IPv6 Security for Low Power and Lossy Networks

Konstantinos Rantos*, Alexandros Papanikolaou†, Charalampos Manifavas‡ and Ioannis Papaefstathiou†

*Technological Educational Institute of Kavala
Kavala, Greece
Email: krantos@teikav.edu.gr

†Technical University of Crete
Chania, Crete, Greece
Email: alxpapanikolaou@gmail.com, ygp@ece.tuc.gr

‡Technological Educational Institute of Crete
Heraklion, Crete, Greece
Email: harryman@epp.teicrete.gr

***Abstract*— Low-power and lossy networks (LLNs) are continually gaining popularity, due to their wide range of applications. Such networks mainly comprise of resource-constrained devices that feature Internet connectivity. This raises many security concerns that cannot be easily resolved, since the inherent limitations of LLN nodes do not allow the direct applicability of existing solutions. This paper proposes an IPsec header compression format for the 6LoWPAN adaptation layer that runs over IEEE802.15.4. The proposed scheme utilises AES-CCM* (CCM-Star), a cryptographic functionality supported by the underlying IEEE802.15.4 protocol, features low packet overhead, and provides both message authentication and confidentiality.**

I. Introduction

Low-power and lossy networks (LLNs) usually comprise embedded systems nodes that are resource-constrained in terms of processing power, energy source, memory size and communications bandwidth. They have become very popular lately, as new applications try to make the most out of their potential, bringing them one step closer to realising the Internet of Things (IoT).

The requirement for Internet connectivity of such devices has raised serious security issues regarding their data communications. Nevertheless, due to their resource-constrained nature, researchers have to come up with new ways for adopting well-established security protocols and communication stacks to such devices, in order to preserve their interoperability.

One way for interconnecting constrained and conventional devices is the use of IETF's 6LoWPAN standards that enable communication over IEEE 802.15.4 networks (referred to as 802.15.4 hereafter), by taking into consideration the various limitations of LLN nodes' resources in terms of computational power, energy consumption and memory size. The said limitations tend to affect the size of the transmitted messages, as well as the type of the cryptographic operations and protocols that can be applied to them. The various header compression formats introduced by the 6LoWPAN standards provide IPv6 connectivity to such nodes, therefore making them part of the web. Nevertheless, having nodes accessible from any location poses serious threats against unauthorised access, which could lead to undesired disclosure and/or manipulation of data. Hence, it is of great importance to construct suitable security mechanisms for protecting such constrained devices, while adhering to the various standards, so as to preserve connectivity and interoperability.

This paper is an extended version of the work presented in [1]. The proposed scheme implements IPsec with Advanced Encryption Standard (AES) [2] in a variant of Counter with CBC-MAC (CCM) mode [3], CCM* (CCM-Star) [4]. By exploiting the functionality of AES-CCM*, it applies it to Encapsulating Security Payload (ESP) [5], thus making it able to provide both confidentiality and message authentication services without the overhead of Authentication Headers (AH) [6]. RFC 4309, however, is not applicable to LLNs and header compression has to be defined in conformance with the formats introduced by 6LoWPAN [7]. We therefore provide a suitable such scheme, able to accommodate the additional information required by AES-CCM*. The choice of AES-CCM* meets the functionality requirements for the offered security services, produces a minimal packet overhead, while the cryptographic operations benefit from the speed and security of AES.

The rest of this paper is structured as follows. In Sec. II we provide some background work, while justifying the use of security mechanisms in IP layer as opposed to the mechanism defined in 802.15.4 for securing data link layer frames, and the use of AES-CCM* mode for ESP. In Sec. III we define the proposed scheme while in Sec. IV we provide the evaluation results of our implementation of the proposed scheme on Contiki OS platform and finally the paper concludes in Sec. V.

II. Securing 6LoWPAN with IPsec

A. Related Work

Securing communications in low power nodes can be applied to several layers of the TCP/IP stack, each having its own advantages and drawbacks (discussed in detail in Sec. II-B). Since this work deals with security in the network layer, the presented related work will focus on solutions proposed and/or implemented for this layer.

An initial feasibility study of the application of IPsec in wireless sensor networks (WSNs) was performed in [8]. Given that IPv6 had been adopted for use on WSNs (as defined by 6LoWPAN) and that IPsec is already part of IPv6, the authors analysed the security and performance trade-offs emerging from the application of cryptography measures in WSNs.

The first implementation of IPsec for 6LoWPAN nodes was presented in [9], [10]. Apart from the available support for most modern algorithms offering data confidentiality and integrity, a header encoding scheme was also proposed, for coping with the small packet size. Our work is based on the aforementioned implementation and it is therefore discussed in more detail later in this paper.

The aforementioned approaches involved "porting" IPsec to 6LoWPAN, while adhering to the latest standards. In this way, secure communication among heterogeneous nodes would be easier to achieve, by using well-established security protocols. Nevertheless, several security architectures for the network layer have also been proposed, mainly for wireless sensor networks (WSNs). ContikiSec [11] is one such case, that was designed for the Contiki OS. Its main aim was to balance low energy consumption and security, while keeping a low memory footprint. Older examples of similar architectures were: The SPINS [12] protocol, part of which was neither fully specified, nor implemented; TinySec [13] that was finally incorporated in the official TinyOS release; SenSec [14], inspired by TinySec and employed more secure crypto algorithms, as well as a resilient keying mechanism; MiniSec [15], another security architecture for TinyOS.

B. IPsec vs 802.15.4 Security

When dealing with communications, security on exchanged messages can be applied to most of the layers of the TCP/IP stack, with the most prominent being the application, network and data link layers. Well-known security mechanisms for these three layers are the TLS [16] protocol and the variant proposed for securing UDP messages, namely DTLS [17], IPsec for the network layer, while for the data link layer, the inherent security mechanism of 802.15.4 [18], assuming that this is the chosen protocol.

Each of these mechanisms has benefits and drawbacks and satisfies different needs. Providing security at the lower layers of the network stack, e.g. 802.15.4 security, relieves applications from employing their distinct security mechanisms for protecting communications, hence simplifying deployment and saving valuable space in resource-constrained nodes. However, this comes at a cost. Security at the data link layer in LLNs introduces significant computational overhead to the nodes as messages are protected at the lowest layer and therefore protection takes place on a node-by-node basis. Hence, messages have to be decrypted, verified and re-encrypted at every single node, typically using a different set of keys, prior to being forwarded to the next node. This process consumes valuable node resources of routing nodes.

IPsec, as opposed to 802.15.4, offers end-to-end protection. Therefore, routing nodes resources are only utilised for message routing, hence saving energy and time. Moreover, IPsec can offer confidentiality, integrity and message authentication to all application data. However, IPsec applied to an LLN cannot provide traffic flow confidentiality the way it does it in IPsec ESP, since it requires the use of tunnel mode to conceal source and destination addresses and padding to conceal communication patterns and characteristics.

Another advantage of using IPsec is that it offers communicating parties the choices among *authentication only*, *encryption only*, or *authentication and encryption*, based on the corresponding communications needs. More specifically, AH can only provide authentication and integrity to exchanged messages, while ESP can provide confidentiality and, optionally, authentication to a reduced set of header fields compared to AH. One of the inherent limitations of AH is that it cannot provide authentication for any mutable fields of the IP header [6]. Namely, fields that may change in transit from source to destination node and their value is not predictable by the sender. The packet overhead for the additional AH functionality is effectively the size of the AH header as defined in [6]; this is equal to 12 bytes plus the size of the Integrity Check Value (ICV), which varies according to the chosen algorithm. Since the scheme proposed in this paper does not make use of AH (its functionality is partially covered by ESP), it will not be discussed any further. The use of AES-CCM* proposed in this paper provides encryption and, optionally, authentication.

Therefore, security at the network layer can be considered as an attractive solution for security communications in low power and lossy networks.

C. Why AES-CCM* ?

Advanced Encryption Standard [2] in Counter with CBC-MAC mode [3], in contrast to other block cipher modes which typically offer only encryption, is an authenticate-and-encrypt mode [3]. It can be used with ESP [5], to provide confidentiality, data origin authentication and integrity [19]. Among the benefits of CCM mode are that it requires minimal message expansion (caused by the Initialization Vector and the Integrity Check Value), it only requires a single key and it can handle messages of arbitrary length, i.e. it does not require padding.

It is worth pointing out that the functionality of offering only authentication is not supported by AES-CCM, according to the original description of the cipher in [3]. This feature was proposed in the version named CCM* (CCM-Star), which is defined in [4] and in effect separates the mechanism of message encryption from that of integrity checking and allows them to be called independently.

In addition, CCM* mode inherits from CCM the very low overhead of only 5 bytes plus the size of the message integrity check (MIC), thus making it a very strong candidate for applications where space limitations exist. CCM* mode is part of the 802.15.4 standard and many of its features have been adopted in this paper.

AES-CCM* is the algorithm chosen by 802.15.4 [18] to secure data link layer frames, and therefore, LoWPAN nodes communicating over 802.15.4 are bound to support it. Moreover, such nodes are also likely to bear hardware modules for AES-CCM* encryption, decryption and integrity check processing. Regardless the type of the 802.15.4 node, i.e. hardware or software, the cryptographic functionality is expected to

be available to upper layers to use it. For this particular reason, many decisions made by the authors regarding the encodings of header fields were affected by the standardised 802.15.4 use of AES-CCM*. This choice might cause problems regarding the interoperability with RFC 4309 [19], which can be bypassed with the use of a bridge mechanism.

D. IPv6 Encoding Header Format

Deploying IPv6 in LLNs running over 802.15.4 [18] poses a major challenge. This is due to the very limited size of an 802.15.4 frame which is restricted to 127 octets[1], therefore not satisfying IPv6 requirement for an MTU of at least 1280 octets. If the frame is secured, the effective payload for the Medium Access Control (MAC) layer is reduced down to 81 octets. Such a limited message length requires special handling in order to transfer IPv6 messages through these channels. As a result, the 6LoWPAN adaptation layer was introduced to act as a bridge between these two layers and reduce the large IPv6 header. The 6LoWPAN solution is header compression and is defined in RFC 6282 [7].

Header compression requires adopting an encoding format, namely LOWPAN_IPHC, for effective size reduction of IPv6 addresses and other IPv6 header fields. LOWPAN_IPHC consists of 2 or 3 octets where the first three bits are the dispatch value, as it is defined in RFC 4944 [20]. It is used to define which IPv6 header fields are elided or "placed immediately after the LOWPAN_IPHC, either in a compressed form if the field is partially elided or literally" [7], as shown in Fig. 1.

The value of the *NH* field included in the LOWPAN_IPHC is used to denote whether the full 8 bits of the next header are carried in line (*NH*=0), or the *Next Header* field is elided (*NH*=1) and the value of IPv6 next header is compressed using the LOWPAN_NHC encoding defined in RFC 6282 [7]. In the latter case the encoded LOWPAN_NHC immediately follows the compressed IPv6 header. The first bits in the LOWPAN_NHC encoding disclose the value of IPv6 *Next Header*.

As a result, an IPv6 datagram following the LOWPAN_IPHC encoding, might contain a mix of compressed and in-line fields and headers, which might be using the LOWPAN_NHC encoding, as shown in Fig. 2.

The encoding of LOWPAN_NHC is depicted in Fig. 3. Note that according to IPv6, next header can either be a Transport Layer protocol header (e.g. TCP, UDP) or an extension header (e.g. IPsec). Looking specifically at IPv6 extension headers, according to RFC 6282 "the LOWPAN_NHC encodings for IPv6 Extension Headers are composed of a single LOWPAN_NHC octet followed by the IPv6 Extension Header" [7].

The format of the LOWPAN_NHC octet for IPv6 extension header is shown in Fig. 4. The *EID* field identifies the IPv6 Extension Header that follows the LOWPAN_NHC byte. EID values 5 and 6 are IANA unregistered values and reserved for future use [7], although proposals have been made to use them for IPsec extension header(s) [9], [10]. *NH* has the same role as previously mentioned and is used to denote whether the full 8 bits of the Next Header, i.e. the extension header, declared by EID are carried in-line (NH=0) or the Next Header field is elided and the next header is encoded using LOWPAN_NHC (NH=1).

[1]The maximum MAC layer header size is 25 octets, hence leaving only 102 octets for the payload. If AES-CCM-128 is used for protecting these messages, this leaves only 81 octets for upper layers. If no compression is used for the IP and UDP headers, hence another 40 plus 8 bytes are needed respectively, only 33 bytes are left for the application layer data.

E. AES-CCM with IPsec ESP

Advanced Encryption Standard (AES) [2], can be used in several block cipher modes, including the Counter with CBC-MAC (CCM) mode [3]. This mode is the choice of preference for 802.15.4 message protection while a scheme to use it with IPsec Encapsulating Security Payload (ESP) is defined in RFC 4309 [19].

AES-CCM* takes as input two parameters:

- **M**: the length in octets of the authentication data, also known as Integrity Check Value (ICV). The ICV is computed for the ESP header, Payload, and ESP trailer fields.
  Although M can take the values of 4, 6, 8, 10, 12, 14, and 16 [3], only values that are multiple of 4 can provide adequate security [19]. RFC 4309 accepts the values of 8 and 16 and optionally 12 octets. 802.15.4 relaxes the lower limit value of M that RFC 4309 sets and accepts the values 0, i.e. authentication is not used, 4, 8 and 16. In this paper both approaches are considered and the use of values 0, 4, 8, and 16 octets is proposed to take into account resource restrictions and allow the absence of an authentication field, should the policy permit it.
- **L**: the size of the length field in octets, of all the encrypted data, including optional ESP Padding, Pad Length, and Next Header Fields. Although CCM defines values of L between 2 and 8 octets, RFC 4309 accepts only the value of 4. This value well exceeds the needs of 802.15.4 where L is set to 2.

For AES-CCM* mode the Payload found in a typical ESP header consists of the Initialisation Vector (8 octets), followed by the Encrypted Payload and the Authentication Data, i.e. an encrypted ICV, both of variable size.

AES-CCM* mode, as previously mentioned and in contrast to ESP, requires no plaintext padding. However, if data is decided to be padded, e.g. for traffic flow confidentiality by concealing the exact message length, it has to be done according to ESP, i.e. to align those data to 32-bit words. Depending on message length, this solution might not be acceptable for the very restricted 802.15.4 frame size. In this case ESP Padding, *Pad Length*, and *Next Header* fields must be concatenated to the plaintext data prior to being encrypted.

III. ESP WITH AES-CCM* ENCODING

A. Available Options

Use of uncompressed IPsec ESP with AES-CCM* in 6LoWPAN over 802.15.4 networks is clearly inefficient due to the large size of the ESP header which 802.15.4 cannot easily accommodate. Such an approach, therefore, requires

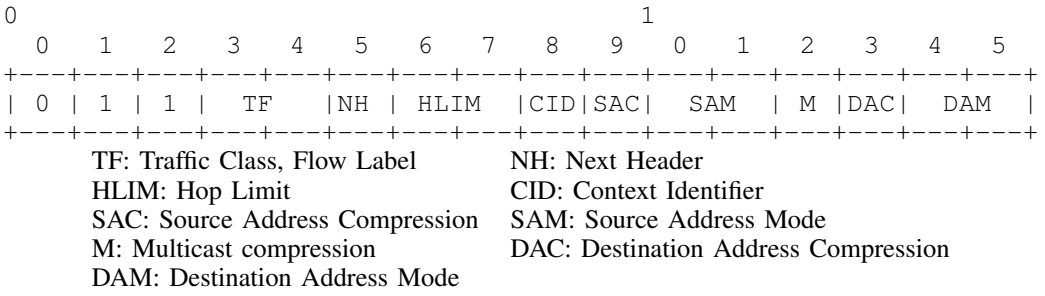

Fig. 1. LOWPAN_IPHC base format.

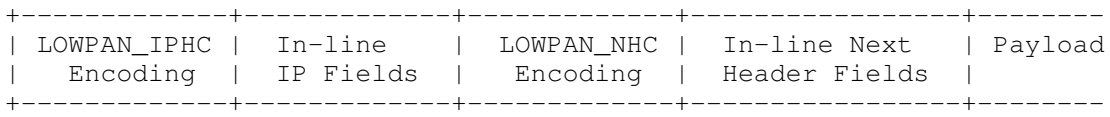

Fig. 2. LOWPAN_IPHC base format.

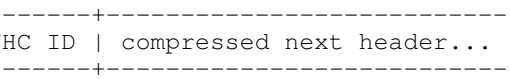

Fig. 3. LOWPAN_NHC Encoding.

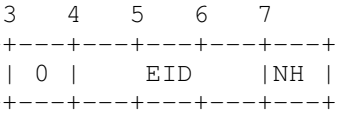

Fig. 4. LOWPAN_NHC octet for IPv6 Extension header.

the encoding of the ESP header using LOWPAN_NHC. The encoding of ESP with AES-CCM* mode is not covered by the encodings proposed in either [9] or [10], since not all required information for the proper use of AES-CCM* can be accommodated in these headers. Therefore, a new encoding is introduced to satisfy the AES-CCM* requirements.

There are typically three options, initially proposed in [10]), to encode the ESP header and use the two EID reserved values.

- One reserved EID value (Fig. 4) is used to denote that an IPsec protocol header is to follow, while the ID bits of the encoded extension header (NHC ID), shown in Fig. 3, define whether next header refers to AH or ESP.
- Two LOWPAN_NHC encodings are introduced for AH and ESP respectively. This is the approach taken in [9] where the authors used both reserved EID values for the new headers. One of the reserved EID values is used to denote that what follows is an AH while the other EID value is used for the ESP header. In this case the NHC ID bits are redundant.
- One reserved EID slot is used followed by one additional NHC for IPsec, which is in turn followed by a compressed AH or ESP header. The special encoding LOWPAN_NHC_IPSEC is used to further introduce another LOWPAN_NHC encoding, one for AH and one for ESP.

B. The Proposed ESP Header

Given that the use of AES-CCM* only applies to ESP and not to AH, the solution proposed in this paper, requires using only one of the available EID values, i.e. value 101, leaving the other reserved for future needs. What follows the LOWPAN_NHC is the LOWPAN_NHC_ESP header defined here. For this purpose, we propose reserving the IANA unassigned values 11000000 up to 11011111 for the LOWPAN_NHC_ octet.

The format of the LOWPAN_NHC_ESP header is shown in Fig. 6.

The LOWPAN_NHC_ESP fields can convey all the necessary information for AES_CCM:

- **SPI (Security Parameter Index)**: If *SPI*=0 the default SPI is used and the *SPI* is omitted. If *SPI*=1 all 32 bits are carried inline, following the ESP header (LOWPAN_NHC_ESP).
- **AI (Address Inclusion)**: AES-CCM* offers the capability to include headers in the computation of the authentication field, without encrypting them. This typically allows the inclusion of extra fields to the payload header in the computation of the ICV. Such fields are the nodes' addresses. If *AI*=1 the addresses are included in the computation of the authentication code, while if *AI*=0 they are omitted.
- **SN (Sequence Number)**: If *SN*=0 then the sequence number required to construct the 13-byte *Nonce* field (see below), is inline and consists of 2 bytes, while the left most 16 bits are assumed to be zero. If *SN*=1 all 32 bits (4 octets) are carried inline after the ESP header (LOWPAN_NHC_ESP).
- **PD (Padding)**: This field is used to denote whether padding is added to the data prior to being encrypted, according to the ESP specifications [5] and RFC 4309 [19]. In contrast to ESP specifications the *Pad Length* field is optional and must only be present if *PD*=1. In this case the padding data must also be present while *Padding*, *Pad Length* and *Next Header* fields must be

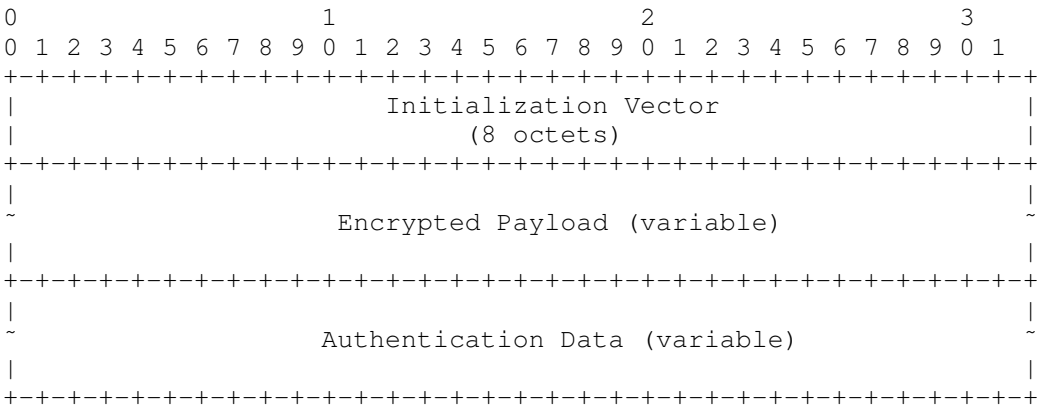

Fig. 5. ESP Payload Encrypted with AES-CCM*.

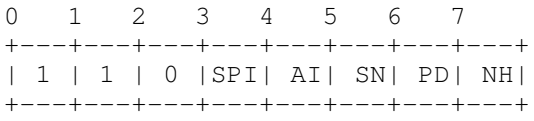

Fig. 6. LOWPAN_NHC_ESP header format.

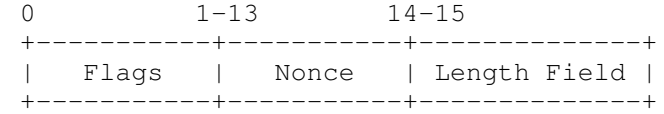

Fig. 7. The first counter block.

concatenated prior to being encrypted, according to RFC 4309 [19].

- **NH (Next Header)**: If *NH*=0, the *Next Header* field in ESP will be used to specify the next header and it is carried inline. If *NH*=1, the *Next Header* field in ESP is skipped. The next header will be encoded using NHC. This is only possible if hosts are able to execute 6LoWPAN compression/decompression and encryption/decryption jointly.

Using AES in counter mode requires generating a sequence of counter blocks, based on an IV carried in each packet [19]. These counter blocks are in turn used to generate the key stream. The AES counter block has a length of 16 octets and comprises of a 1-byte *Flags* field, a 13-byte *Nonce* and a 2-byte *Length* field (shown in Fig. 7) is the first block that has to be constructed for authentication purposes.

The *Flags* field is in turn formatted as shown in Fig. 8.

The 1-bit long *Reserved* field is reserved for future expansions and shall be set to 0. The 1-bit *Adata* field is set to 0 if there is no authentication data. The *M* field is the 3-bit representation of the integer $(M-2)/2$ if $M > 0$ (where M is the length of the authentication field in octets, i.e. 0, 4, 8 or

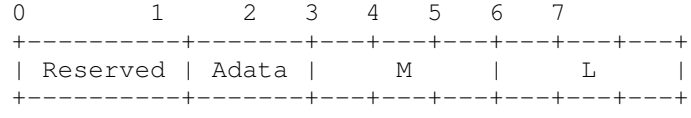

Fig. 8. Flags byte.

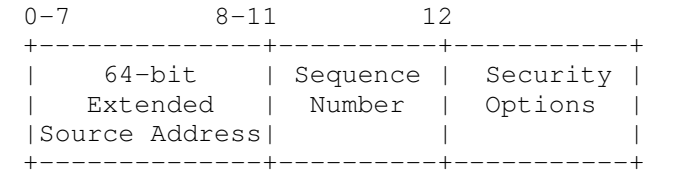

Fig. 9. The 13-byte Nonce field.

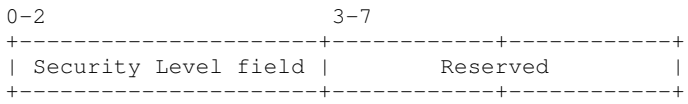

Fig. 10. Security Options byte.

16) and of the integer 0 if $M = 0$, in most-significant-bit-first order [18]. The *L* field, i.e. the length of the *Message Length* field in octets, is the 3-bit representation of the integer $L-1$, in most-significant-bit-first order [18]. Recall that the value L should be set to 2.

The 13-byte *Nonce* field consists of the fields shown in Fig. 9.

The *Sequence Number* field is used to specify the identifier for the frame. The *Security Options* byte comprises the 3-bit long *Security Level Field* and a 5-bit long *Reserved* field.

C. Offered Security Levels

The *Security Level* field, inline with the 802.15.4 standard can take the values shown in Table I, to denote whether encryption is used and the size of the Message Integrity Check (MIC). Note that security level 0 must not be accepted as a valid option as it denotes total absence of protection. In addition, security level 4 is not recommended to be used in real-life scenarios, as the lack of an integrity-checking mechanism makes it vulnerable to replay attacks, similar to the ones described in [21], [22].

AES-CCM* messages remain secure as long as the same nonce N is not used more than once with the same K. Using a 4-byte frame counter in the Nonce field allows using the same key for up to 2^{32} encrypted messages from the same source. Re-keying should therefore be considered during the node's lifetime based on the frequency of the exchanged messages.

TABLE I. SECURITY LEVEL FIELD.

Security Level	Security Level field	Security Attributes	Data Confidentiality	Data Authenticity	Encrypted authentication tag length, M octets
0	000	None	No	No	0
1	001	MIC-32	No	Yes	4
2	010	MIC-64	No	Yes	8
3	011	MIC-128	No	Yes	16
4	100	ENC	No	No	0
5	101	ENC-MIC-32	Yes	Yes	4
6	110	ENC-MIC-64	Yes	Yes	8
7	111	ENC-MIC-128	Yes	Yes	16

All computations that follow the construction of the corresponding fields considering the encodings defined in this paper, should follow the process defined in 802.15.4. As previously mentioned, this design approach was taken to ensure that implementations benefit most from the cryptographic functionality found in 802.15.4 supporting nodes, i.e. AES-CCM* support.

D. Modes of Use

There are different scenarios with distinct characteristics regarding the use of IPsec in LLNs.

1) Node-to-node within the same network. The proposed schemes allow any two nodes, such as the sink node and any other node, to communicate securely using IPsec, hence hiding the communicated data from intermediate routing nodes.
2) Node to remote party without gateway. The remote party has to support the compressed IPsec ESP format using AES-CCM* as proposed in this paper.
3) Node to remote party through gateway. The remote party has to support the IPsec ESP mode using AES-CCM* as defined in RFC 4309 [19], while the gateway, supporting both RFC 4309 and the scheme proposed in this paper, needs to transform these messages accordingly prior to forwarding them to the other end.

In the first and the third case the advantages are that addresses can be completely elided, since they can be fully-derived from the link-local address.

IV. EVALUATION

A. Experimental Setup

We implement IPsec with AES-CCM* using the Contiki operating system [23] and the COOJA simulator [24], using Tmote Sky motes for the simulation setup. The IPsec headers were compressed for 6LoWPAN in the manner presented in Sec. II-E and Contiki's μIP stack was modified accordingly. The AES implementation used was the one provided by the MIRACL library [25].

B. Estimated Memory Usage

Tmote Sky motes have a Texas Instruments MSP430 microcontroller, with 10 KB RAM and 48 KB Flash memory [26]. Therefore, the `msp430-size` utility of the respective compiler toolchain can be used for providing an estimate of the expected memory usage, both in terms of flash memory and stack (RAM) size. The obtained values were approximately 45.5 KB for flash and 8 KB for RAM, thus allowing the scheme's deployment on most available resource-constrained platforms.

TABLE II. MEMORY USAGE ESTIMATES ACCORDING TO THE OUTPUT OF `msp430-size`.

Implementation	Memory Estimate (KB)	
	Flash	SRAM
With S/W AES	45.5	8.0
With H/W AES	42.9	7.8
Difference	2.6	0.2

It is also worth emphasising that COOJA is unable to simulate code that utilises any AES implementation on the mote's hardware (in the case of the Tmote Sky, such functionality is provided by the CC2420 chip), thus requiring a software implementation as well. We therefore compiled the code twice: The first time including the MIRACL's AES implementation and the second time using only the necessary statements that utilise the provided functionality of the CC2420 chip. By subtracting the two, we can calculate the memory required by the software AES implementation. Hence, the flash memory estimate is larger by approximately 2.6 KB, as it includes the object code of MIRACL's AES implementation. These figures are summarised in Table II.

C. Packet Overhead

In order to get an idea of the imposed packet overhead, the proposed scheme is compared to compressed IPsec that uses the "traditional" approach of AH and ESP, as well as to 802.15.4. It should be emphasised that 802.15.4 supports only link-layer security, which has the advantage of lower packet overhead, at the expense of increased power consumption, as has already been mentioned in Sec. II-B. The main difference among these three schemes is that the ones using IPsec are able to offer end-to-end security, whereas the 802.15.4 inherent link-layer security can only offer node-to-node security.

The packet overhead among the compared three schemes, using various configurations, is presented in Table III. Calculations have been performed under the assumption of optimal compression where applicable and perfect block alignment (namely, no padding takes place). The compressed IPsec version with AH and ESP proposed in [9] targets 6LoWPAN networks and the 802.15.4 itself is for resource-constrained devices, therefore forming good candidates for comparison to the proposed scheme.

When only message authentication is required, 802.15.4 features the least overhead, equal to the size of the MAC (12

TABLE III. PACKET OVERHEAD (IN BYTES) FOR IPSEC WITH AES-CCM*, COMPARED TO SIMILAR APPROACHES FOR 6LOWPAN NETWORKS, ASSUMING OPTIMAL COMPRESSION AND ABSENCE OF PADDING, IF APPLICABLE.

Security Service	**Proposed Scheme**		**Compressed IPsec**		**802.15.4**	
	Attributes	Overhead	Attributes	Overhead	Attributes	Overhead
Authentication	MIC-32 MIC-64 MIC-128	10 14 24	AH with HMAC-SHA1-96	16	AES-CBC-MAC-96	12
Encryption	ENC	12	AES-CBC	12	AES-CTR	5
Both	ENC-MIC-32 ENC-MIC-64 ENC-MIC-128	10 14 24	AH with HMAC-SHA1-96 and ESP with AES-CBC	24	AES-CCM-128	21

bytes). The extra headers required for IPsec increase the total overhead to 16 bytes for the compressed IPsec scheme. The proposed scheme has the same overhead for the authentication-only options (an equivalent of 18 bytes in total for a 96-bit long MIC, although this option is not supported).

For providing solely message encryption through IPsec increases the total overhead to 12 bytes for both schemes, compared to the mere 5 bytes of 802.15.4, which uses AES-CTR.

If both encryption and authentication are required, the compressed IPsec implementation has an additional 3 bytes of packet overhead compared to the 802.15.4 approach. Nevertheless, the proposed scheme is more efficient than the compressed IPsec one, as it uses an 128-bit long MIC instead of a 96-bit long one. This is due to the use of AES-CCM*, which offers an authentication mechanism along with the encryption mechanism, without requiring any additional information. This feature does not hold in the case where AH and ESP are combined, as each individual component requires its own set of headers, thus introducing additional packet overhead.

D. Cryptographic Performance

The security levels (Table I) were benchmarked for variable input sizes, in order to draw conclusions regarding their performance and therefore their applicability to scenarios involving resource-constrained devices.

As can be seen from Fig. 11, there are virtually no differences among the MIC-only options. This is due to the fact that the MAC is truncated from its fully-computed version (128 bits), thus performing in full the most computationally-expensive part and simply skipping the processing of the last few bytes. It is worth pointing out that the calculated MIC gets encrypted before it is output.

The ENC-MIC options exhibit similar characteristics, as they essentially are MIC-only options with an additional overhead equal to the encryption/decryption operations of the full plaintext/ciphertext, respectively.

The ENC-only option is clearly the fastest of all, since AES in Counter mode requires far less processing than its CBC counterpart. Nevertheless, its use is not recommended, as it has already been mentioned in Sec. III-C.

According to [24], COOJA's MSPSim module can perform very accurate measurements regarding the CPU duty cycle. Given that the CPU energy consumption on a Tmote Sky mote is 5 mW, it is possible to calculate the expected consumption

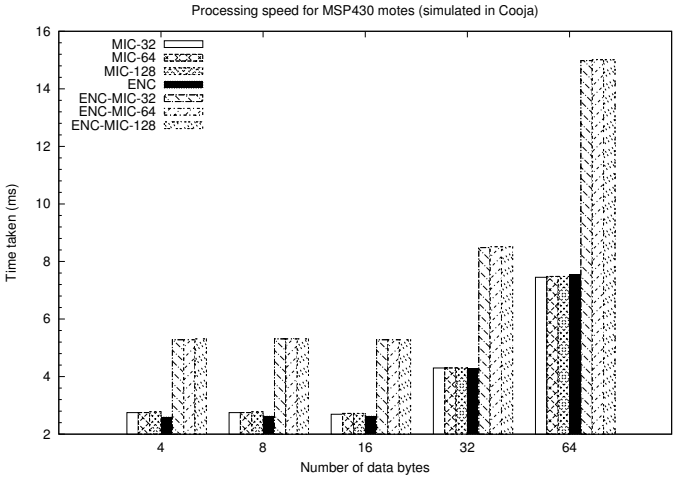

Fig. 11. Processing speed measurements for Tmote Sky motes, simulated in COOJA.

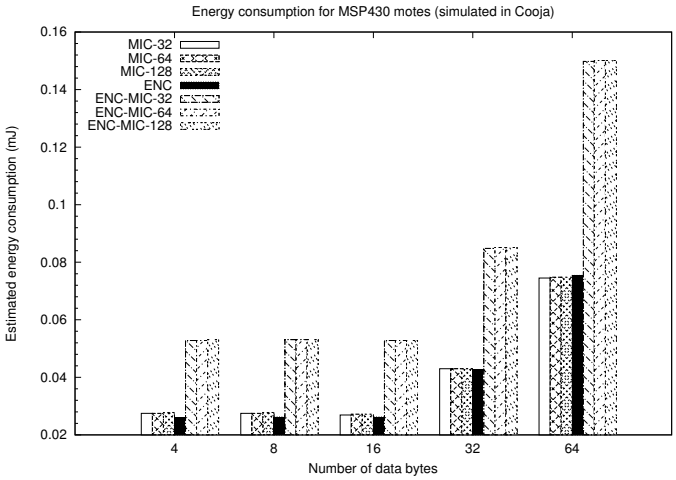

Fig. 12. Estimated energy consumption for Tmote Sky motes, simulated in COOJA.

of the cryptographic operations for the given mote, with high accuracy. The estimated CPU energy consumption has therefore been computed and is presented in Fig. 12.

It should be noted that measurements on real hardware did not take place. However, exploiting the hardware cryptographic abilities of the e.g. CC2420 chip that a Sky mote has, it should yield significant speed-ups, similar to the ones mentioned

in [9].

V. Conclusions

Efforts towards realising the Internet of Things (IoT) involve the use of resource-constrained devices in the form of low-power and lossy networks (LLNs). However, the widespread use of such devices, combined with the requirement for Internet connectivity, face serious security threats of unauthorised access and/or manipulation of the communicated data. One of the possible security mechanisms in the IP layer is IPsec, which can offer end-to-end encryption, thus avoiding repeated decryptions and re-encryptions of transmitted data in a hop-by-hop manner, as in the case of the link layer security mechanism of IEEE 802.15.4. Nevertheless, given the packet size limitations of IEEE 802.15.4 (127 bytes), any extensions must be made sparingly, so as to ensure the availability of sufficient data space for the upper layers.

In this paper we proposed a new encoding format for IPsec ESP protocol applicable to LoWPAN networks, coupled with a suitable IP header compression scheme, to ensure compliance with the 6LoWPAN specifications. AES in CCM* mode was selected as the security mechanism for providing the various security services, due to its ability to offer both data confidentiality and authenticity, without requiring any additional data in the form of extension headers. In this way, the packet overhead is smaller than similar schemes, which combine AH and ESP for offering the same security services.

Future work involves examining the applicability of the proposed scheme in heterogeneous networks, while adhering to the latest standards, to ensure compatibility and interoperability.

Acknowledgment

This work has been supported by the Greek General Secretariat for Research and Technology (GSRT), under the ARTEMIS JU research program nSHIELD (new embedded Systems arcHItecturE for multi-Layer Dependable solutions) project. Call: ARTEMIS-2010-1, Grand Agreement No.: 269317.

References

[1] K. Rantos, A. Papanikolaou, and C. Manifavas, "IPsec over IEEE 802.15.4 for low power and lossy networks," in *11th ACM International Symposium on Mobility Management and Wireless Access (MobiWac)*, Barcelona, Spain, 3–8 November 2013, to appear.

[2] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Advanced Encryption Standard (AES)," NIST, FIPS PUB 197, 2001.

[3] D. Whiting, R. Housley, and N. Ferguson, "Counter with CBC-MAC (CCM)," RFC 3610, 2003.

[4] R. Struik, "Formal specification of the CCM* mode of operation," IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs), Tech. Rep., 2005.

[5] S. Kent, "IP Encapsulating Security Payload (ESP)," RFC 4303, 2005.

[6] ——, "IP Authentication Header," RFC 4302, 2005.

[7] J. Hui and P. Thubert, "Compression format for IPv6 datagrams over IEEE 802.15.4-based networks," RFC 6282, 2011.

[8] J. Granjal, J. Sa Silva, E. Monteiro, J. Sa Silva, and F. Boavida, "Why is IPSec a viable option for wireless sensor networks," in *5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS 2008)*, Atlanta, GA, USA, 29 September – 2 October 2008, pp. 802–807.

[9] S. Raza, S. Duquennoy, T. Chung, D. Yazar, T. Voigt, and U. Roedig, "Securing communication in 6LoWPAN with compressed IPsec," in *7th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '11)*, Barcelona, Spain, 2011, pp. 1–8.

[10] S. Raza, T. Voigt, and U. Roedig, "6LoWPAN extension for IPsec," in *Interconnecting Smart Objects with the Internet Workshop*, Prague, Czech Republic, 2011.

[11] L. Casado and P. Tsigas, "ContikiSec: A secure network layer for wireless sensor networks under the Contiki operating system," in *Identity and Privacy in the Internet Age*, ser. LNCS, A. Jøsang, T. Maseng, and S. J. Knapskog, Eds. Berlin, Heidelberg: Springer, October 2009, vol. 5838, pp. 133–147.

[12] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar, "SPINS: Security protocols for sensor networks," in *7th Annual International Conference on Mobile Computing and Networking*, Rome, Italy, 16–21 July 2001, pp. 189–199.

[13] C. Karlof, N. Sastry, and D. Wagner, "TinySec: A link layer security architecture for wireless sensor networks," in *2nd International Conference on Embedded Networked Sensor Systems (SenSys 2004)*, Baltimore, MD, USA, 03–05 November 2004, pp. 162–175.

[14] T. Li, H. Wu, X. Wang, and F. Bao, "SenSec design," I^2R Sensor Network Flagship Project (SNFP: Security part), Technical Report TR v1.0, February 2005.

[15] M. Luk, G. Mezzour, A. Perrig, and V. Gligor, "MiniSec: A secure sensor network communication architecture," in *6th International Conference on Information Processing in Sensor Networks (IPSN 2007)*, Cambridge, MA, USA, 25–27 April 2007, pp. 479–488.

[16] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) protocol version 1.2," RFC 5246, 2008.

[17] E. Rescorla and N. Modadugu, "Datagram Transport Layer Security version 1.2," RFC 6347, 2012.

[18] IEEE Std 802.15.4, 2011.

[19] R. Housley, "Using Advanced Encryption Standard (AES) CCM mode with IPsec Encapsulating Security Payload (ESP)," RFC 4309, 2005.

[20] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 packets over IEEE 802.15.4 networks," RFC 4944, 2007.

[21] S. M. Bellovin, "Problem areas for the IP security protocols," in *Sixth Usenix Unix Security Symposium*, 1996, pp. 205–214.

[22] K. G. Paterson and A. K. L. Yau, "Cryptography in theory and practice: The case of encryption in IPsec," in *25th Annual International Cryptology Conference (Eurocrypt 2006)*, ser. Lecture Notes in Computer Science, vol. 4004, Saint Petersburg, Russia, 2006, pp. 12–29.

[23] A. Dunkels, B. Grönvall, and T. Voigt, "Contiki – A lightweight and flexible operating system for tiny networked sensors," in *1st IEEE Workshop on Embedded Networked Sensors (Emnets-I)*, Tampa, USA, 2004.

[24] J. Eriksson, F. Österlind, N. Finne, N. Tsiftes, A. Dunkels, T. Voigt, R. Sauter, and P. J. Marrón, "COOJA/MSPSim: Interoperability testing for wireless sensor networks," in *2nd International ICST Conference on Simulation Tools and Techniques (SIMUTools '09)*, Rome, Italy, 2009, pp. 27:1–27:7.

[25] CertiVox, "Multiprecision Integer and Rational Arithmetic C/C++ Library (MIRACL)," https://github.com/CertiVox/MIRACL, visited 2013-06-06.

[26] Moteiv Corporation, "Tmote Sky – Ultra low power IEEE 802.15.4 compliant wireless sensor module (datasheet)," http://www.snm.ethz.ch/snmwiki/pub/uploads/Projects/tmote_sky_schematic.pdf, 13 November 2006.