Network-layer security for the Internet of Things using TinyOS and BLIP

Jorge Granjal, Edmundo Monteiro, Jorge Sá Silva
University of Coimbra
{jgranjal,edmundo,sasilva@dei.uc.pt}

***Abstract*—The design of standard communications and security mechanisms for resource-constrained sensing applications and devices may provide an important contribution for its integration with the Internet and consequently towards the realization of what we nowadays identify as the Internet of Things (IoT). Strong security assurances will be required for many applications that are expected to manipulate and transmit sensitive data using wireless communications. Security mechanisms should thus be designed and adopted for the IoT that are both standard and flexible. Standardization enables the widespread adoption of compatible security solutions, while flexibility may guarantee that security mechanisms may be easily adapted to a wide range of heterogeneous sensing devices and applications, as we expect to encounter in the IoT.**

In this paper we target our work on the design and experimental evaluation of security mechanisms for communications at the network-layer with sensing devices (smart objects) using the standard IPv6 protocol. It is certain that not all smart objects on the IoT will have the capability or be required to support IPv6, but we nevertheless believe that the availability of secure end-to-end communications at the network layer with other sensing devices or with Internet hosts may enable a much richer integration of sensing applications with the Internet. It may also enable new types of sensing applications where smart objects are able to cooperate remotely and securely using Internet communications.

Our work proposes and evaluates the usage of new compressed security headers for the network layer with smart objects. We implement and evaluate what is, as far as we know, the first implementation of security at the network layer experimentally evaluated using the TinyOS operating system and its BLIP networking stack. As we verify in the course of our evaluation study, various scenarios employing network-layer secure communications involving smart objects are feasible, particularly when security mechanisms are designed to benefit from cross-layer interactions that allow the optimization of expensive cryptographic operations.

***Index Terms*—Internet of Things, smart objects, 6LoWPAN, compressed security headers, TinyOS, BLIP**

I. Introduction

The 6LoWPAN (IPv6 over Low Power Personal Area Networks) group of the IETF was mandated with the task of designing an adaptation layer that enables the transmission of IPv6 packets over IEEE 802.15.4 [1] networks. Despite this initial focus on a particular technology, other communications standards such as Bluetooth Low Energy (BLE) or Power Line Communication (PLE) are expected to be supported in the future, allowing a myriad of heterogeneous sensing and actuating devices to communicate using standard IP protocols. This aspect may certainly contribute to evolution of the current Internet architecture towards something we currently identify as the Internet of Things (IoT), an Internet where communications with sensing devices and applications are supported and transparent. Other than the adaptation layer, the 6LoWPAN group has also defined mechanisms such as neighbor discovery and address auto-configuration that allow a sensing device to activate its presence on an existing IPv6 network of smart objects. The 6LoWPAN group has produced two RFC documents to date, RFC 4919 [2] discussing general goals and assumptions of the group and RFC 4944 [3] describing the adaptation layer and related header compression mechanisms. As we analyze throughout the paper, header compression is omnipresent in all 6LoWPAN solutions, given the extremely limited payload space to transmit data using LoWPAN (Low Power Wireless Personal Area Networks) technologies such as IEEE 802.15.4.

Although the successful integration of 6LoWPAN networks with the Internet will require security to be properly addressed from the start, we note that it has not been properly addressed in 6LoWPAN, as only generic considerations and recommendations [4] have been produced so far, but not any specific mechanism to enable security in the context of the adaptation layer. There is therefore no current solution to enable secure end-to-end communications with IPv6-enabled smart objects using the adaptation layer, probably due to the assumption that security will be addressed in other layers. We must realize that in practice 6LoWPAN enables many interesting usage scenarios, for example with two smart objects on remote locations communicating in the context of a distributed sensing application, or when an Internet host is able to obtain information directly from a sensing device. Therefore, as network-layer security was designed as a cornerstone of the current Internet, it may also play an important part in the context of broader secure integration architecture for the IoT.

We previously pioneered the idea of enabling security at the network layer for 6LoWPAN smart objects [5], and more recently proposed a secure interconnection model [6][7] where such mechanisms are theoretically validated. The current paper describes the implementation and the experimental evaluation of such mechanisms, considering its requirements of vital resources on resource-constrained smart objects.

The paper proceeds as follows. In Section II we analyze related work, and Section III describes how the proposed compressed 6LoWPAN security headers can be employed. Section IV describes the experimental evaluation setup used to validate our proposal and in Section V we analyze the results obtained from the experimental evaluation study. In Section VI we analyze the overall effectiveness of 6LoWPAN security and Section VI concludes the paper.

II. Related work

Current proposals to implement secure end-to-end communications between smart objects and Internet hosts mostly target the transport layer, in particular by proposing modified versions of the SSL (Secure Sockets Layer) protocol. For example SSNAIL [9] proposes a light-weighted version of SSL to be supported by Internet hosts and smart objects. Other proposals do exist that only provide partial end-to-end security such as Sizzle [10], which employs SSL to secure communications between an Internet host and a security gateway protecting the network of smart objects from the Internet, with such communications being translated to a proprietary communications protocol in the network of smart objects. Another research proposal can be found in ContikiSec [11] which, although providing security at the link layer only, introduces the idea of employing different security modes that can be related to the security requirements of a particular sensing applications or device. Table 1 illustrates the main characteristics of these research proposals.

TABLE I
RESEARCH PROPOSALS FOR SECURITY AT HIGH LAYERS WITH SMART OBJECTS

	SSNAIL	Sizzle	ContikiSec
Authentication	ECC (ECDSA)	ECC (ECDSA)	CMAC
Key negotiation	ECC (ECDH)	ECC (ECDH)	Not supported
Key size(s)	160 bits	160 bits	128 bits
Data encryption	RC4	RC4	AES
Hashing/Integrity	MD5, SHA1	MD5, SHA1	CMAC
Access control	Not supported	Security Gateway	Not supported
Operational layer	Transport (SSL)	Transport (SSL)	Link-layer
Gateway usage	No	Yes	No
End-to-end security	Yes, with SSL	Yes, with SSL	Not supported

These proposals have shown that security can be effectively employed at higher communication layers with resource constrained smart objects, something that is in deep contrast with the classic perception of many researchers. Nevertheless, two important aspects are missing from these proposals that we believe are vital for security in the context of the IoT, and can be (at least partially) answered by network-layer security mechanisms. One is that security mechanisms should be available that provide security for communications independently of the applications. In this respect, SSL presents the limitation of requiring explicit support from sensing applications. Other relevant aspect is that security mechanisms should be adaptable to the characteristics and security requirements of particular sensing applications. Regarding this aspect, mechanisms that work with fixed configurations in terms of parameters that control its security and resource usage are not appropriate for the IoT. Aspects such as the cryptographic algorithms employed and relevant configuration parameters such as cryptographic key size and frequency of key refreshment deeply influence the lifetime of sensing applications and resource-constrained devices. Security mechanisms should therefore allow the establishment of acceptable compromises between resources required for performing security operations and the security level required for a particular sensing application. Other than the proposals described in Table I, another proposal also exists to enable security at the 6LoWPAN layer [22] with the same goal as ours, but that nevertheless suffers from diverse limitation that we strive to address in the current work. Authors consider the usage of context-based header compression to enable 6LoWPAN security, a compression technique currently not accepted as standard. Other limitations of this proposal can be found on its validation, as it doesn't consider the usage of security in tunnel and transport modes when evaluating its impact on payload space, neither of the usage of variable-sized keys and authentication data, two important requirements for the adaptability of security to sensing applications with different security requirements. Authors also do not perform a proper study on the impact of the proposal on the lifetime of sensing applications, an aspect that is vital when evaluating the effectiveness of any security proposal for resource-constrained sensing devices.

It is our belief that security can be integrated at the 6LoWPAN adaptation layer with the characteristics previously identified as desirable, and thus enabling the usage of application-independent and flexible security mechanisms which can play an important part in the integration of smart objects with the Internet. As security at the network-layer is certainly not a solution to all security issues and will not be appropriate for extremely restricted devices such as Radio-Frequency Identification (RFID) devices, the security architecture adopted for the IoT must also target other necessary security mechanisms.

III. Security in the 6LoWPAN adaptation layer

We begin by discussing security in the context of 6LoWPAN header compression mechanisms, a fundamental concept of the adaptation layer and that must therefore be considered during the design of new security mechanisms.

A. Security in the context of header compression

One major goal of the 6LoWPAN adaptation layer is the support of fragmentation and reassembly of IPv6 packets transmitted over LoWPANs. This is a necessity because IPv6 determines that any communications link may be able to support a minimum MTU of 1280 bytes, while the MTU of LoWPANs is typically lower. For example, with IEEE 802.15.4 only 102 bytes are available (without link-layer security) of payload space, as Figure 1 illustrates. The payload space available when using IEEE 802.15.4 depends on the overhead introduced by addressing and control information at the link layer. As IEEE 802.15.4 also provides security at the link layer [1], its usage also influences the payload space available at the end.

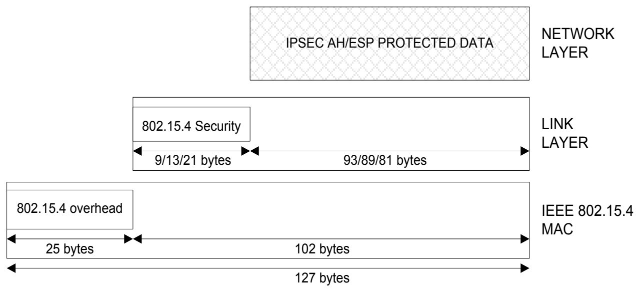

Figure 1. Payload space available for 6LoWPAN using IEEE 802.15.4 considering also the usage of link-layer security

As illustrated in the previous Figure, IEEE 802.15.4 provides three link-layer security modes. The AES-CCM-128 security mode requires 21 bytes of payload space, AES-CCM-64 requires 13 bytes and AES-CCM-32 requires 9 bytes. We are considering the usage of an IEEE 802.15.4 auxiliary security header occupying 5 bytes, with 1 byte being used for the security control field and 4 bytes for the frame counter field, also considering that the cryptographic keys required for security are obtained automatically from the source and destination link-layer addresses of the frame [1]. As network-layer security can protect communications even for data transmitted in a wireless network of smart objects, for the purpose of the evaluation of our proposal later in the paper we consider the availability of 102 bytes as the data payload for 6LoWPAN, meaning that we dispense link-layer security. As link-layer security is available at the hardware in many sensing devices, the fact that link-layer security mechanisms are not activated doesn't mean that such efficient encryption and authentication mechanisms can't be of use. We consider the design of cross-layer security mechanisms for the 6LoWPAN adaptation layer, which allow us to benefit from the availability of such efficient cryptographic operations as we discuss later with our proposal. Figure 1 also illustrates the reason why header compression is so prevalent in 6LoWPAN, as even when not using link-layer security applications do not have that much space left to transmit data.

B. Compressed security headers for 6LoWPAN

As IEEE 802.15.4 doesn't provide any type of multiplexing information to allow a receiver to distinguish among different types of data packets, 6LoWPAN uses the first byte of the link-layer payload as a dispatch byte which allows the identification of the transported packet and (if necessary) further information within the subtype. We need therefore to decide how new headers for security are going to be identified in 6LoWPAN using the dispatch byte. Three strategies would allow us to identify the presence of new headers in the context of 6LoWPAN, as we proceed to discuss.

The first option is to use the ESC header type value [3] which allows the usage of an additional dispatch byte to identify the presence of new headers. Using this approach the first (original) dispatch byte remains untouched and the following (new) dispatch byte can be used to identify new security headers. This approach presents the inconvenience of requiring one additional byte for this purpose. A second option is to use context-based header compression as in [22], particularly using the LOWPAN_IPCH and LOWPAN_NHC headers, and to define appropriate identification values for security using the EID field of the LOWPAN_NHC header. This approach presents the limitation that context-based header compression is not currently adopted as standard [15], and thus a better option would be to integrate security in the context of standardized headers and identification values. The final and the option we adopt consists in the definition of new dispatch type values for security using reserved values of the original payload byte, as RFC 4944 [3] describes. We proceed by describing how such identification values are defined.

1) New 6LoWPAN dispatch type values for security

6LoWPAN uses the first two bits of the dispatch byte (the first byte of the IEEE 802.15.4 payload) to allow nodes to identify the presence of a 6LoWPAN packet or of other types of packets. For a 6LoWPAN packet, the remaining bits of the dispatch byte allow the identification of specific types of 6LoWPAN headers that correspond to given functionalities of the adaptation layer, namely a mesh, fragmentation or addressing header. When the first two bits identify a 6LoWPAN addressing header (value '01', please refer to Table II), several dispatch values are reserved as RFC 4944 [3] describes. We use four values from the set of reserved values to identify the presence of new 6LoWPAN compressed security headers and respective usage modes, as Table II describes. It is important to note that the usage of reserved dispatch values is both accepted and encouraged in RFC 4944, which defends that with the further development of 6LoWPAN additional functions are expected to occupy unused space [3].

TABLE II
NEW DISPATCH VALUES TO IDENTIFY 6LOWPAN SECURITY AND USAGE MODES

Header dispatch values for 6LoWPAN security	6LoWPAN security header and usage mode
01 001xxx	AH in transport mode
01 010xxx	AH in tunnel mode
01 011xxx	ESP in transport mode
01 100xxx	ESP tunnel mode

The values in the previous Table are more precisely obtained from the set of reserved values after LOWPAN_HC1, which is the value defined to identify the presence of a HC1 compressed addressing header. HC1 is the header compression format adopted in 6LoWPAN to compress addressing information, while HC2 was defined to allow the compression of transport-layer UDP header information. As we can see in the previous Table, the first 3 of the remaining 6 bits of the dispatch byte are sufficient to identify a security header, together with its usage mode and irrespective of the value of the reaming 3 bits. The 3 remaining bits by it selves are sufficient to distinguish between different types of 6LoWPAN addressing headers. This identification strategy therefore gives us the possibility of simultaneously identifying the presence of security and addressing information on a given 6LoWPAN packet, allowing also to save payload space and easing the processing of headers in tunnel and transport modes.

2) *Compressed ESP header for 6LoWPAN*

The design of new security headers for the 6LoWPAN adaptation layer must take into consideration several aspects. The first is that the principles of simplification, compression and shared context around which other 6LoWPAN headers [3] were designed should also be considered for security. At the same time, it is desirable that the processing of such headers can be easily integrated into existing implementations of the Internet Security architecture [14], as this would contribute to its evolution towards easily adopting new IPv6-enabled sensing applications. Another important aspect is that most sensing platforms currently possesses or will probably adopt in the future hardware cryptographic operations. Hardware encryption and authentication must therefore be considered together with cryptographic algorithms implemented in software. For example, IEEE 802.15.4 requires hardware cryptography and platforms such as the TelosB [20] mote support hardware security with the AES cryptographic algorithm in CCM* combined mode [14] using the cc2420 chip. AES/CCM provides encryption and decryption in the CTR (Counter) mode and authentication and integrity in the CBC-MAC mode. Considering also that AES/CCM is part of the set of future mandatory algorithms for the Internet Security architecture, we realize the importance of its consideration during the design of security headers for 6LoWPAN.

In Figure 2 we illustrate how the 6LoPWAN ESP [12] security header is build, and in the same Figure we also illustrate which fields are integrity protected (with an 'I') and encrypted (or confidentiality) protected (with a 'C'). The purpose of this header is to provide applications with encryption and optional authentication and integrity of 6LoWPAN packets.

By analyzing the 6LoWPAN ESP header illustrated in Figure 2, we begin by identifying a 2-byte SPI (Security Parameters Index) field, whose purpose is to allow a receiving entity to relate an incoming packet to a specific security association. This allows a sensing device to obtain information such as the cryptographic algorithms and keys required to apply security operations to the packet. The next field stores a 2-byte sequence number, with the purpose of helping end systems in protecting against packet replay attacks. The sequence number is treated as an unsigned value and implementations must ensure that a distinct value is maintained for each different security association. Given the transmission rates of typical sensing applications, 2 bytes is considered appropriate for this field. Nevertheless, if necessary an option similar to ESN (Extended Sequence Numbers) [12] may be designed for 6LoWPAN, allowing communicating parties to agree on larger sequence numbers. A 6LoWPAN ESN option would allow a device to maintain a larger sequence number for security associations requiring it, with such number being used for ICV-computation purposes, while only its lower 2 bytes are transmitted with each 6LoWPAN packet. It is important to note in this context that other than the usage of a distinct sequence number for each security association, a key management mechanism appropriate to 6LoWPAN must be designed to allow keys to be periodically refreshed. This is necessary because algorithms as AES/CCM completely loose its security if a given key is ever reused with the same sequence number.

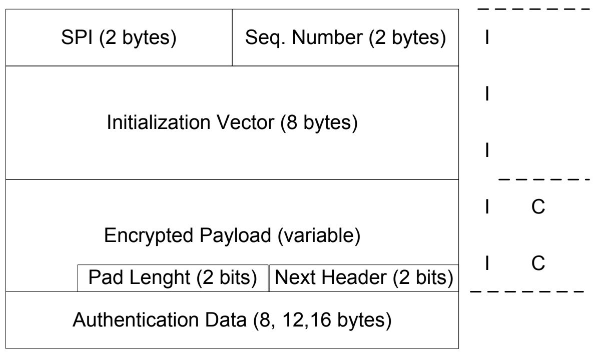
SPI (2 bytes)
Seq. Number (2 bytes)
Initialization Vector (8 bytes)
Encrypted Payload (variable)
Pad Lenght (2 bits)
Next Header (2 bits)
Authentication Data (8, 12,16 bytes)
I
I
I
I
C
I
C

Figure 2. Compressed ESP security header for 6LoWPAN

After in the ESP header we encounter the IV (Initialization Vector) field, which is used to transport cryptographic synchronization data necessary for two devices to successfully apply the same cryptographic algorithm. Synchronization data is used as input to 3DES and AES in CBC (cypher-block chaining mode) algorithms or together with additional data generated by end devices to produce the input required for algorithms such as AES in CTR (counter) mode. CRT mode is available with hardware implementations of AES/CCM, and the rules currently defined for the usage of AES in CRT mode with the ESP header [16] state that 3 bytes of salt must be added to the IV data for this purpose, since AES requires an 11-byte nonce. Again, by following such rules we allow an easier integration of our new 6LoWPAN security headers in current implementations of the Internet Security architecture.

Next in the packet comes the encrypted data, at the end of which two fields are added that aid in employing the security header with different encryption algorithms and usage modes. The first is the pad length field, which stores the number of padding bytes (from 1 to a maximum of 4) added to the original encrypted data to align up the payload and trailer, if required by the encryption algorithm employed. Next appears the next header field, which stores information on how the receiver should interpret the decrypted data by indicating the presence of a TCP, UDP or ICPMv6 packet.

At the end of the 6LoWPAN ESP header follows the ICV (Integrity Check Value) or MIC (Message Integrity Code) field, which stores the authentication data used to authenticate the origin of the 6LoWPAN packet and verify its integrity. As such operations are optional with the ESP header they are only performed if required in the context of a given security association. The size of the authentication data depends on the encryption algorithm used to generate the MIC code and on the level of integrity and authentication required for the given security association. This field is of 12 bytes if generated using HMAC-SHA1-96 or AES-XCBC-MAC-96, since both algorithms produce a 96-bit MIC code. When using hardware AES/CCM, this algorithm can be used to generate 8, 12 or 16 bytes MIC codes, also in line with recommendations from RFC 4309 [16]. The MIC code is not protected by encryption, meaning that smart objects are able to verify the authenticity and integrity of a received 6LoWPAN packet protected with ESP before being required to perform more computational demanding decryption operations.

The layout of the 6LoWPAN ESP header in the previous Figure was decided considering also the previously discussed requirements for the design of 6LoWPAN headers in general. Compression and simplification are performed whenever possible, while at the same time the relevant fields are appropriately dimensioned for its usage with software and hardware cryptographic algorithms. The same principles were observed during the design of the 6LoWPAN authentication header, which we describe next.

3) *Compressed AH header for 6LoWPAN*

The purpose of the 6LoWPAN authentication header is to allow end systems that do not require confidentiality to verify the integrity and origin of a given 6LoWPAN network-layer packet, and also to provide protection against replay attacks. The usage of the authentication header is interesting as such operations are less demanding of resource-constrained smart objects than encryption and decryption. Many sensing applications on the IoT will probably do not require encryption, as the data transported is itself not confidential, while the most important will be to protect communications against corrupted packets and to authenticate its origin. In Figure 3 we illustrate the 6LoWPAN compressed AH (authentication header), and in the same Figure we also indicate which parts of the header and data payload are integrity and authentication protected (as indicated by an 'A') and are considered mutable (as indicated by an 'M') or immutable fields (as indicated by an 'I') in respect to the computation of the ICV. Mutable field are fields for which the sending device (which must compute the ICV) is unable to calculate or predict its final value upon arrival of the packet at its destination, and thus such values are zeroed for IVC computation purposes. An added advantage of the 6LoWPAN authentication header over its ESP counterpart is that security (authentication and integrity) can also be applied to fields outside the security header itself. The authentication data is computed considering all the fields identified as immutable in Figure 3 (with the ICV field itself being zeroed for that purpose), but implementations may also decide to include immutable fields of the HC1 or HC2 addressing and transport headers. The padding required by the integrity algorithm (if any) and the high-order bits of the ESN option (if adopted for 6LoWPAN in the future, as previously discussed) are also considered during the computation of the ICV.

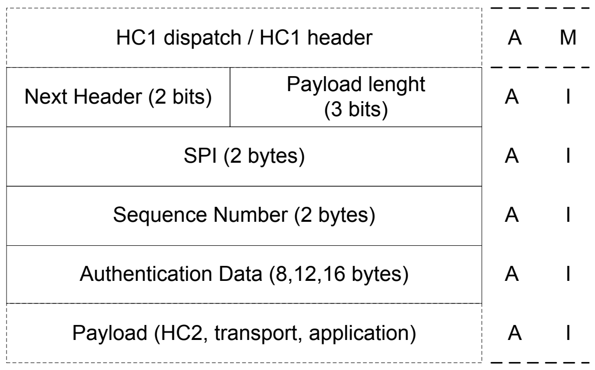
HC1 dispatch / HC1 header
A M
Next Header (2 bits)
Payload lenght (3 bits)
A I
SPI (2 bytes)
A I
Sequence Number (2 bytes)
A I
Authentication Data (8,12,16 bytes)
A I
Payload (HC2, transport, application)
A I

Figure 3. Compressed AH security header for 6LoWPAN

Analyzing the 6LoWPAN AH header illustrated in the previous Figure, the next header field allows the identification of the next header as TCP, UDP or ICMPv6. The 3-bit payload length field stores the total length of the header in units of 32-bit words. The SPI field again allows a device to map the 6LoWPAN packet to a particular security association, and the sequence number supports protection against packet replay attacks. The size of the authentication field is proportional to the integrity and authentication level required for the security association and is in line with the set of cryptographic algorithms that can be used for its generation, as was previously discussed for the ESP header and utilized in our experimental evaluation study later in the paper.

As a final remark concerning the 6LoWPAN security headers described, one aspect to note is that they don't allow the maintenance of nice 32-bit or 64-bit boundaries that were a concern during the design of its counterparts for the Internet Security architecture. This is not so much of a problem for 6LoWPAN because it is designed to be implemented in sensing platforms employing 8-bit or 16-bit microcontrollers.

4) *Usage of security in the context of existing 6LoWPAN headers*

As other 6LoWPAN headers are currently defined, we need to consider how the new 6LoWPAN security headers are allowed to be employed side-by-side with them. We need therefore to analyze the usage of security together with the mesh addressing header, the fragmentation header and the compressed addressing header. This is important not only in respect to the implementation of a security-enabled 6LoWPAN networking stack, but also because it allows us to investigate the impact of security on the final payload space available to 6LoWPAN applications, as we do in our evaluation study later in the paper.

The mesh addressing header transports information for layer-two forwarding whenever a mesh-routing protocol is employed for routing packets from node to node in the LoWPAN. It is important to note that mesh-routing is independent of 6LoWPAN, as IPv6 only cares about the source and destination addresses of the devices, independently of how the packet arrives at its destination. The fragmentation header transports information related to how the original IPv6 packet was fragmented for its transportation in the 6LoWPAN, and which therefore is necessary for the reassembly of the original packet at the destination node. Finally, the compressed addressing header allows the compression of IPv6 addresses and multicast addresses whenever possible.

Considering that 6LoWPAN security is inherently end-to-end, meaning that it is intended to be generated and interpreted by 6LoWPAN devices, the headers that are destined to be interpreted by each device on the path of the 6LoWPAN packet towards its final destination must not be considered for security purposes. This applies to the mesh addressing header, which therefore must appear before any 6LoWPAN security header independently of its usage mode. The same rationale can be applied to a broadcast (LOWPAN_BC0) and fragmentation headers. A broadcast packet stores a sequence number intended to be interpreted at each forwarding node, allowing the implementation of the broadcast mechanism using a flooding communications algorithm. The fragmentation header transports information necessary for the reassembly of the IPv6 packet at the 6LoWPAN destination. In summary, we consider that 6LoWPAN security headers protect only end-to-end payloads as makes sense for network-layer security, and as such appear after the mesh, broadcast and fragmentation headers.

As with the traditional Internet Security architecture, we consider that 6LoWPAN security may be useful in two different usage modes, the tunnel mode and the transport mode. Transport mode enables secure communications between two end devices (smart object or other type of 6LoWPAN or IPv6 device) and will be preferred in many usage scenarios, also considering that it requires less header space from the (already limited) link-layer payload. On the other end, tunnel mode allows for the tunneling of secure communications via intermediate devices functioning as security gateways or as 6LoWPAN routers. The usage of 6LoWPAN security in these two usage modes is discussed in greater detail next in the paper.

5) Tunnel and transport mode usage scenarios

In Figure 4 we illustrate the usage of 6LoWPAN security in transport mode, side-by-side with other compressed 6LoWPAN headers and data from transport protocols and applications. As previously discussed, the mesh, broadcast and fragmentation headers (when present) appear before security headers. Security is identified side-by-side with compressed addressing, and in transport mode acts on the payload of the original packet, which may contain an HC2 compressed UDP header and data from other transport protocols and applications. The scope of this 6LoWPAN security header in transport mode depends on it being AH or ESP, as previously discussed. When using authentication and integrity, a security trailer (MIC or ICV) it is transmitted at the end.

Mesh + BC0 + Frag type	Mesh + BC0 + Frag header	HC1 + sec disp.	HC1 header	Security header	Payload (HC2, transport, application)	Sec. Trailer /ICV

Figure. 4. Usage of 6LoWPAN security in transport mode

Regarding the usage of 6LoWPAN security in tunnel mode, two addressing headers are necessary and security is employed as we illustrate in Figure 5. The inner addressing header identifies the address of the ultimate destination of the 6LoWPAN packet, which may be for example a 6LoWPAN smart object, while the outer addressing header identifies the immediate (intermediate) destination of the packet, for example a security gateway placed between the Internet and the network of smart objects supporting a given sensing application, or a 6LoWPAN router supporting secure communications between remotely deployed sensing devices.

Mesh + BC0 + Frag type	Mesh + BC0 + Frag header	HC1 disp.	HC1 header	HC1 + sec disp.	HC1 header	Security header	Payload (HC2, transport, application)	Sec. Trailer /ICV

Figure. 5. Usage of 6LoWPAN security in tunnel mode

The usage of 6LoWPAN security in tunnel mode allows the protection of the entire inner (ultimate) addressing header and also of the original data payload. As previously discussed, which fields are considered for security depends on the usage of AH or ESP, and may also depend on particular implementations of 6LoWPAN security, as networking stacks may decide to include information from compressed transport headers such as HC2 for example. As with security in transport mode, authentication data follows at the end if necessary.

IV. EXPERIMENTAL EVALUATION SETUP

The validation of any proposal on security for resource-constrained sensing devices is of particular relevance if performed experimentally, as in practice several unpredicted aspects related to the functioning of sensing devices and wireless communications are difficult to reproduce realistically using simulation environments. We proceed by describing the experimental evaluation scenario and discussing conclusions obtained from our experimental measurements.

A. Experimental evaluation scenario

Our proposal on security for the 6LoWPAN adaptation layer was implemented using the TinyOS operating system, more precisely by modifying its BLIP networking stack. For the experimental measurement of the values relevant for our evaluation study we employed UDP communication sessions established between different 6LoWPAN devices, in particular between a TelosB [20] mote and a Linux host supporting both 6LoWPAN and IPv6. The Linux host is a router between an Ethernet IPv6 network and the IEEE 802.15.4 LoWPAN, employing a second TelosB mote as a bridge supporting communications with the network of smart objects. This Linux 6LoWPAN router also supports the RADV daemon which implements routing advertisements for the 6LoWPAN. The TelosB mote used for the experimental evaluation of the measured parameters is a battery-powered device supporting our TinyOS testing application and the 6LoWPAN security-capable BLIP networking stack. The TelosB mote is currently a popular research platform providing a good reference for the validation of our proposal, as it is a good representative of the computational power currently available with commercial sensing platforms. In particular, the TelosB is powered by a 16-bit RISC MSP 430 microcontroller with 10Kbytes of RAM for program execution and 48Kbytes of ROM for program storage. It also supports communications at 2.4GHz and data transmissions at 250Kbps. Because it implements IEEE 802.15.4 standard, it also provides hardware encryption and authentication using the AES/CCM cryptographic suite, which we consider in our proposal and employ during our experimental evaluation study.

B. *Identification of appropriate cryptographic algorithms*

The selection of the cryptographic algorithms that are appropriate to support 6LoWPAN security and to the resources of smart objects is an important requirement for our experimental study. Such algorithms or suites of algorithms enable smart objects to perform encryption, decryption and integrity/authentication related operations, therefore allowing the processing and generation of information transported with 6LoWPAN using security headers. For the identification of the appropriate cryptographic suites our goal is in fact twofold, as on the one side the usage of algorithms that are already accepted in the Internet Security architecture would facilitate the integration of sensing applications with the Internet in a secure fashion, while on the other side we must carefully consider the effectiveness of the usage of such algorithms in resource-constrained smart objects. The selection of cryptographic algorithms regarding its impact on smart objects must nevertheless not be too conservative in this respect, as it may be expected that sensing devices will become more powerful and energy-efficient in a near future [2], and thus security mechanisms that have been showed to be unviable or marginally viable in the present may well be employed in a more mainstream fashion using future sensing platforms.

As the current Internet Security architecture [14] may evolve to include 6LoWPAN sensing applications in the future, we find it useful to analyze the effectiveness of the usage of the cryptographic algorithms currently defined as mandatory in smart objects. The same applies to the algorithms that will probably be mandatory in the future with the same security architecture. The fact that the Internet Security architecture allows end systems to agree on security algorithms and related security configuration parameters at the establishment of a security association is also in line with our requirement of adaptability for 6LoWPAN security. Adaptable security mechanisms at the network layer may allow a 6LoWPAN smart object to select a cryptographic algorithm from a pool of alternatives and to decide how to use that algorithm, and this serves our goal on providing security mechanisms that allow the establishment of acceptable compromises between security and resources required from smart objects, two aspects we consider important for the IoT. In Table III we identify the cryptographic algorithms that are either currently defined as mandatory for the Internet Security architecture or that will probably be adopted as such in a near future [17].

TABLE III
CRYPTOGRAPHIC ALGORITHMS ADOPTED AS MANDATORY FOR THE INTERNET SECURITY ARCHITECTURE

Security Header	Cryptographic algorithm	Usage	Status
ESP	3DES-CBC	Encryption	Mandatory
	AES-CBC	Encryption	Future
	HMAC-SHA1-96	Authentication	Mandatory
	AES-XCBC-MAC-96	Authentication	Future
	AES-CCM	Combined	Future
AH	HMAC-SHA1-96	Authentication	Mandatory
	AES-XCBC-MAC-96	Authentication	Future

As we can see in Table III, a shift is expected to take place towards AES-based cryptographic solutions. This is also in line with the fact that AES is already supported by various sensing platforms, and also motivated our decision on considering the usage of AES/CCM during the design of the 6LoWPAN security headers. Although AES/CCM is a combined mode cypher, its CCM* mode implemented in platforms such as the TelosB allows for the separate support of integrity, authentication and confidentiality operations as required for the suites that employ AES in the previous Table.

As the usage of standard security and communications mechanisms may facilitate the secure integration of sensing applications with the Internet, our experimental evaluation study considers the usage of the algorithms in Table III in obtaining network-layer 6LoWPAN security. In Table VI we describe how the above algorithms are employed in support of security using the compressed ESP and AH 6LoWPAN headers.

TABLE IV
USAGE SCENARIOS OF CRYPTOGRAPHIC ALGORITHMS AND 6LOWPAN SECURITY HEADERS

Cryptographic suites	6LoWPAN header	Security provided
3DES-CBC	ESP	Confidentiality
AES-XCBC-MAC-96		Integrity, authentication
3DES-CBC	ESP	Confidentiality
HMAC-SHA1-96		Integrity, authentication
AES-CBC	ESP	Confidentiality
AES-XCBC-MAC-96		Integrity, authentication
AES-CBC	ESP	Confidentiality
HMAC-SHA1-96		Integrity, authentication
AES/CCM (HW)	ESP	Confidentiality, integrity, authentication
AES-XCBC-MAC-96	AH	Integrity, authentication
HMAC-SHA1-96	AH	Integrity, authentication
AES/CCM (HW)	AH	Integrity, authentication

As the previous Table reflects, the isolated testing of the algorithm described in Table III would not be appropriate to evaluate the effectiveness of 6LoWPAN security, as in most deployments at least two algorithms will need to be supported, one providing confidentiality (throughout encryption and decryption) and the other providing authentication and integrity (throughout generation and verification of a MIC code or secure hash). AES/CCM was tested as available at the hardware in the TelosB mote, while the other algorithms were programmed in software using code optimized for small microcontrollers with the characteristics of the MSP 430.

The cryptographic block size and key size used with each algorithm are the values inherent of each cryptographic algorithm itself, and are also in line with the configurations required by the Internet Security architecture. Such values constitute therefore the most appropriate configuration to measure the effectiveness of our proposal. In particular, 3DES-CBC uses 128-bit keys to process 64-bit blocks. AES-CBC, AES-XCBC-MAC-96 and AES/CCM (in hardware) use 128-bit keys to process 128-bit blocks. HMAC-SHA1-96 uses 160-bit keys to process 512-bit blocks, with the original 160-bit authenticator generated being truncated to 96 bits, as specified in RFC 2404 [18]. Our AES-CBC software implementation also supports the AES-XCBC-MAC-96 algorithm, with the XCBC mode modifying the classic CBC mode as documented in RFC 3566 [19].

The fact that our tests employ software and hardware based cryptographic algorithms allows us to analyze the feasibility of 6LoWPAN security for a broader set of devices. This is relevant also if we consider that the IoT will include heterogeneous sensing devices which may or may not support hardware security. In the evaluation study we describe next we consider the usage of ESP to provide confidentiality together with authentication and integrity. Although we could have considered using ESP only for confidentiality, we believe that authentication and integrity are security properties that will be required for most of the applications in the IoT. In fact, the opposite may be truer, in that many applications will probably be able to dispense confidentiality and use only AH with its authentication and integrity assurances.

V. Experimental evaluation of 6LoWPAN security

Our evaluation on the feasibility of 6LoWPAN security begins by analyzing its impact on 6LoWPAN payload space. Later in the paper we concentrate on aspects such as its energy and computational requirements, which are determinant for the achievement of acceptable transmission rates and lifetimes for sensing applications.

A. Overhead of security on 6LoWPAN payload space

As the payload space available to applications is an important factor in dictating the usefulness of 6LoWPAN in real usage scenarios, we start by analyzing the packet overhead of the usage of security in both tunnel and transport modes. We start by analyzing the payload space required for 6LoWPAN in various addressing compression scenarios and also with mesh and fragmentation headers. We must also consider the payload space required for the security headers previously described. The space required for such 6LoWPAN headers is described in Table V. The values illustrates in this Table are used during our following analysis on the impact of security on 6LoWPAN payload space.

TABLE V
PAYLOAD SPACE REQUIREMENTS FOR 6LOWPAN ADDRESSING, MESH, FRAGMENTATION AND SECURITY

Scenario	Payload requirement
Link-local unicast	7 bytes
Outside of link-local scope	23 bytes
Outside of local LoWPAN	31 bytes
6LoWPAN AH	37 bits
6LoWPAN ESP	96 bits
Fragmentation	4 bytes / 5 bytes
Mesh addressing	5 bytes / 17 bytes

The first 3 lines of the previous Table refer to the possible address compression scenarios that 6LoWPAN allows. With link-local unicast communications between 6LoWPAN smart objects sharing the same local link address, HC1 and HC2 6LoWPAN compression allows the compression of an UDP/IPv6 header down to 7 bytes. In this scenario the version, traffic class, flow label, payload length and next header fields, and also the link-local prefixes of the IPv6 source and destination addresses are all elided, with the correspondent IPv6 suffixes being derived from the IEEE 802.15.4 header. In the second compression scenario, corresponding to communications with an object outside of the local link while on the same 6LoWPAN, the IID (Interface Identifier) suffix of the source and destination addresses is also obtained from IEEE 802.15.4 addressing information, but the source and the destination prefixes must be carried inline, at the end requiring an additional 16 bytes for addressing information. The third scenario is also the most useful in the context of the IoT, as in this case a 6LoWPAN-enabled smart object is able to communicate directly with an Internet host or with another remote smart object. In this scenario, 6LoWPAN is only able to elide the source address IID, with the remaining part of the source address and with the full destination IPv6 address carried inline, requiring in total 31 bytes.

The other lines in Table V refer to the payload space required for the remaining 6LoWPAN headers, including security. Regarding 6LoWPAN security, from the representations in Figures 2 and 3, and without considering the transportation of encrypted and authentication data, we can see that the authentication header requires 37 bits in total, with the ESP header requiring 96 bits.

Finally, a complete evaluation of the overhead of security on 6LoWPAN payload space must also consider the (optional) usage of mesh and fragmentation information. Fragmentation requires 4 bytes for the first fragment and 5 bytes for subsequent fragments, while the mesh addressing header required 5 or 17 bytes, depending on the usage of short (16-bit) or long (EUI-64 64-bit) addresses, respectively. Such values are also represented in Table V are considering for the following analysis on the impact of 6LoWPAN security on packet payload space.

We begin by illustrating in Figure 6 the impact on the 6LoWPAN payload space of security in tunnel and transport modes, considering the addressing compression scenarios previously discussed and also the transportation of MIC codes (authentication data) of 8, 12 and 16 bytes in length. This Figure illustrates the number of bytes available for 6LoWPAN applications, in percentage of the maximum of 102 bytes available with IEEE 802.15.4 without link-layer security. For comparison purposes, we also illustrate the payload space available when using 6LoWPAN headers without any security-related header or data.

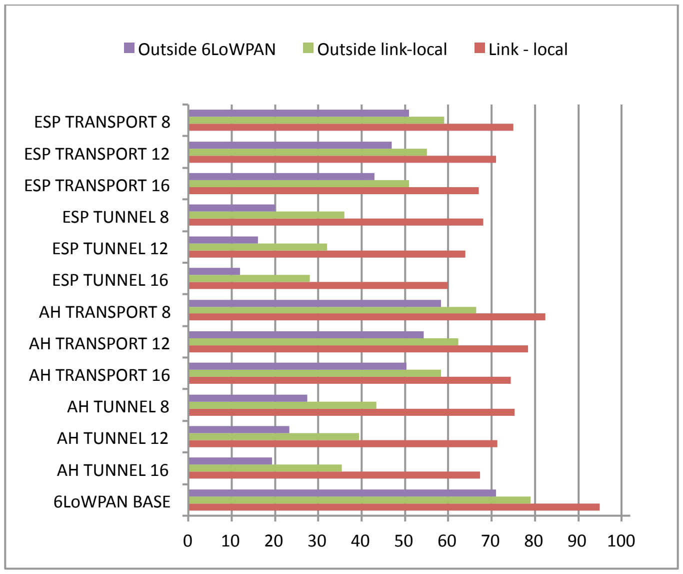

Figure 6. Payload space available with 6LoWPAN security without mesh or fragmentation headers

As in this comparison we don't consider mesh and fragmentation information, the values illustrated in the previous Figure correspond in practice to the maximum payload space that applications can use without requiring 6LoWPAN fragmentation. As we considered the values from Table V for each of the three addressing compression scenarios, the payload space required for HC1 and HC2 compressed headers is already accounted for. As can be seen in Figure 6, transport mode security is clearly less expensive than tunnel mode security in terms of the payload space required. For communications with smart objects on the same local link or with systems outside of the local link but on the same 6LoWPAN, security leaves from 51 to 82 bytes to 6LoWPAN applications using ESP or AH in transport mode. When communications with the outside of the 6LoWPAN are required, the available space also in transport mode is between 43 and 58 bytes. 6LoWPAN security in transport mode therefore provides acceptable availability on payload space, regardless of the security header employed and of the integrity and authentication level required.

Regarding tunnel mode, in reality security in this mode is only useful in support of communications with devices outside of the local link or outside of the 6LoWPAN. In the former scenario, the 6LoWPAN payload available is between 28 and 43 bytes, while on the later it is between 12 and 27 bytes. Tunnel mode security when communications are between devices on different LoWPANs is viable mainly for applications requiring moderate amounts of data. For communications with devices outside of the 6LoWPAN tunnel mode is viable but only for applications requiring the transportation of only a few bytes. This confirms the big impact of tunnel mode security on payload space, especially considering that in this evaluation scenario we are not yet considering the usage of mesh or fragmentation information.

Our next evaluation adds the usage of a fragmentation header, and the obtained values are illustrated in Figure 7. Fragmentation is necessary when a IPv6 packets needs to be transmitted that requires more space than the limit values identified in Figure 6, and in such situation the adaptation layer fragments the packet and inserts a fragmentation header in each 6LoWPAN packet. As the overhead imposed from the fragmentation header is only of 5 bytes per 6LoWPAN packet, our previous conclusions remain valid regarding security in transport mode, as the payload space remains between 38 and 77 bytes. As for tunnel mode security, it leaves between 23 and 38 bytes for communications with nodes outside the local link, and between 7 and 22 bytes for communications with other 6LoWPAN or IPv6 hosts.

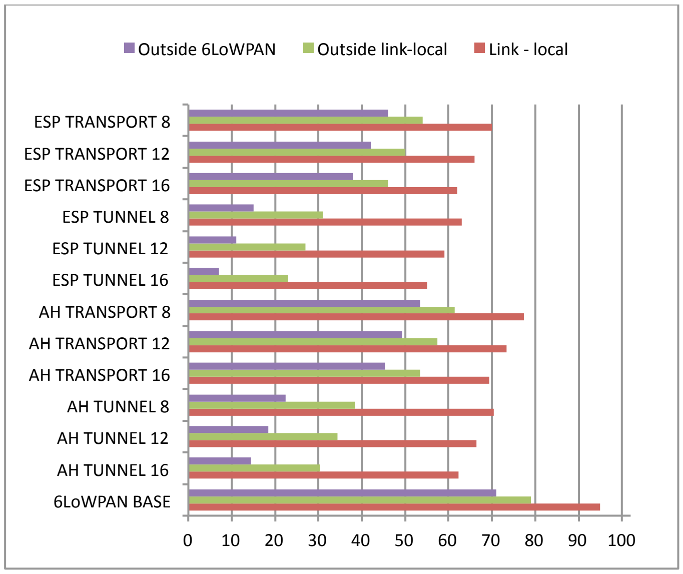

Figure 7. Payload space available with 6LoWPAN security plus fragmentation information

We are therefore able to realize that for tunnel mode the space required for the fragmentation header poses an extra pressure on the usefulness of this security mode. ESP in tunnel mode can only be considered viable if employed with a MIC code with 8-byte or 12-byte. In conclusion, transport mode security remains viable for all compression scenarios. Tunnel mode security is valid for communications with nodes outside of the local link for applications requiring the transmission of small amounts of sensing data, while for communications with Internet hosts it is viable mainly for applications that don't require confidentiality and therefore are able to use AH to protect the transportation of small amounts of data. For applications that do require confidentiality, ESP is only a viable choice if lower integrity and authentication assurances are acceptable, more precisely using ESP with a MIC code with 12 or (preferably) 8 bytes.

Next we consider the usage of a mesh addressing header together with security, and the impact on 6LoWPAN payload space is illustrated in Figure 8. We considered the usage of a mesh addressing header with 17 bytes, what corresponds to mesh addresses being obtained from EUI-64 addresses of the 6LoWPAN sensing devices.

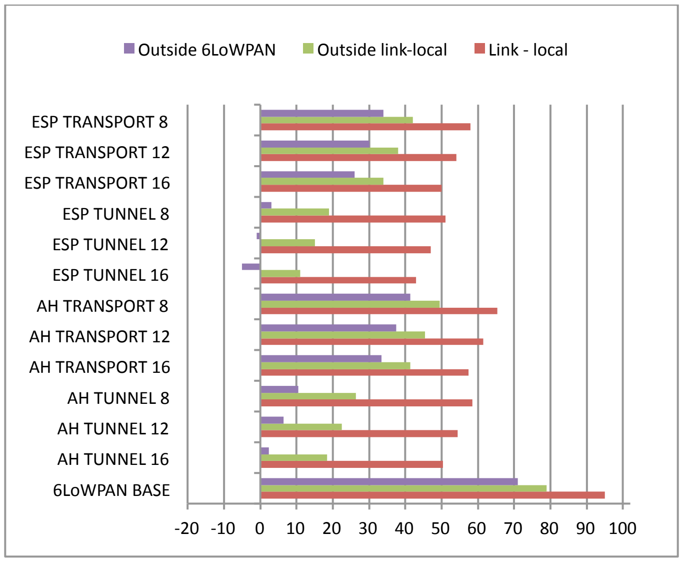

Figure 8. Payload space available with 6LoWPAN security plus mesh information

With mesh addressing and security in transport mode, the payload space available for 6LoWPAN applications drops to between 26 and 65 bytes. As for tunnel mode security, communications with nodes outside of the local link remain possible if small amount of data are required to be transmitted, as in this case only from 11 to 26 bytes are available. For communications with nodes outside of the 6LoWPAN, tunnel mode is viable only with AH transporting MIC codes with 8 bytes, which even so only provides 10 bytes of payload space. The remaining tunnel security usage modes do not provide enough payload space, or the support of 6LoWPAN security headers would require the availability of more than 102 bytes. In conclusion, in the presence of a mesh header security in transport mode security remains valid but only for applications requiring the transmission of a moderate amount of data. Tunnel mode security is viable only for applications requiring low integrity and authentication assurances or which do not need confidentiality at all.

Our final analysis considers the simultaneous usage of mesh and fragmentation headers, which corresponds therefore to the worst usage scenario for 6LoWPAN security in terms of its impact on payload space. In Figure 9 we illustrate the payload space available to applications after the applications of 6LoWPAN security with mesh and fragmentation headers. The values illustrated in this Figure corroborate some of our previous conclusions. We are able to conclude that transport mode security remains a valid usage mode for small amounts of data, as in this case between 21 and 60 bytes are available to transport data from 6LoWPAN applications. Tunnel mode is clearly the most affected mode by the lack of available payload space, and in practice can be considered enviable for communications with nodes outside of the 6LoWPAN, since in this case even AH with a MIC code of 8 bytes would only leave 5 bytes of data payload space, which may be insufficient for most sensing applications on the IoT. We can see that with several configurations there is not enough space to accommodate even just the 6LoWPAN headers.

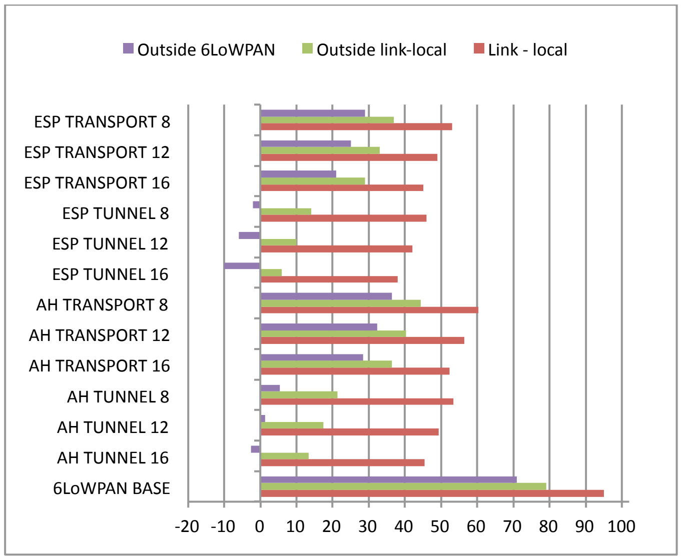

Figure 9. Payload space available with 6LoWPAN security plus fragmentation and mesh information

Regarding tunnel mode communications with nodes outside of the local link, it still can be considered viable for very small amounts of transmitted data, as between 6 and 21 bytes are available. This is especially true for applications that only require authentication and integrity, as with AH in tunnel mode between 13 and 21 bytes are available.

During our previous analysis on the impact of security on 6LoWPAN payload space we were able to identify several viable usage modes. On a broader sense, we need to identify what are the viable usage modes of security that will in fact be useful in the context of future IoT applications. Such usage modes are described in Table VI from the perspective of communications initiated by a 6LoWPAN smart object.

TABLE VI
USAGE SCENARIOS FOR 6LOWPAN SECURITY IN THE CONTEXT OF THE IOT

To / From	6LoWPAN device on same local link	6LoWPAN device outside of local link	Device outside the 6LoWPAN
6LoWPAN device	AH/ESP in transport mode	AH/ESP in transport mode AH/ESP in tunnel mode via 6LoWPAN router	AH/ESP in transport mode AH/ESP in tunnel mode via security gateway

The previous Table considers the usage of two types of 6LoWPAN routers, one acting as a 6LoWPAN security gateway and the other as a 6LoWPAN router. A security gateway is a device without the resource constraints that are typical of smart objects, and that as such can be used to aid in the integration and interconnection of a network of smart objects with the Internet. Such a device may implement various security mechanisms to protect the network of smart objects from the Internet, among which the processing of network-layer security in communications with Internet devices and smart objects. A 6LoWPAN router can be a more limited device supporting distributed sensing applications and allowing routing and enforcing security mechanisms for communications between different 6LoWPANs. Considering also our previous evaluation of the impact of 6LoWPAN security on payload space, in Table VII we resume the main characteristics of the usage modes of 6LoWPAN security that can be identified as viable. In this classification viability means that enough payload space is left for 6LoWPAN applications while guaranteeing the usage of strong authentication codes. It is clear that, without employing a mesh routing protocol, 6LoWPAN network-layer security is viable in all usage modes as long as applications are able to adapt to the payload space available.

TABLE VII
VIABLE USAGE MODES OF NETWORK-LAYER SECURITY

							Available payload space			MIC code size		
	Mesh	Frag	Transport	Tunnel	AH	ESP	High	Medium	Low	16	12	8
Link-local			●		●	●	●			●		
		●	●		●	●	●			●		
	●		●		●	●		●		●		
Outside local link			●		●	●	●			●		
				●	●	●		●		●		
		●	●		●	●	●			●		
		●		●	●	●			●	●		
	●		●		●	●		●		●		
	●			●	●	●			●	●		
	●	●	●		●	●			●	●		
	●	●		●	●				●	●		
Outside 6LoWPAN			●		●	●	●			●		
				●	●	●			●	●		
		●	●		●	●	●			●		
		●		●	●				●	●		
		●		●		●			●		●	
	●		●		●	●		●		●		
	●			●	●				●			●

The classification in Table VII traduces a qualitative evaluation for which preference is given to the usage of strong authentication and integrity codes whenever possible. Other practical usage scenarios can nevertheless be identified to be viable for the IoT, for example considering that some applications may only need to use smaller authentication codes or use ESP without authentication and integrity.

B. *Memory footprint of 6LoWPAN security*

As memory is also a limited resource on smart objects, our evaluation study proceeds with the analysis of the memory footprint of our implementation of 6LoWPAN security in TinyOS and BLIP, while supporting the cryptographic suites previously identified. Different versions of a base TinyOS application were employed in our experimental evaluation, supporting the security-enabled 6LoWPAN stack together with each of the cryptographic suites implemented in software or available in hardware. We separately measure the RAM and ROM memory necessary with each version of the testing application, as both types of memory are limited on the TelosB mote.

In Figure 10 we describe the memory footprint of 6LoWPAN security with each of the cryptographic suites. The values illustrated in this Figure are in percentage of the total of RAM and ROM memory available on the TelosB (10Kbytes of RAM and 48Kbytes of ROM). For comparison purposes, we also evaluate and illustrate the memory required for a base BLIP networking stack with support for the processing of 6LoWPAN security headers but without any cryptographic algorithm. This base application allows us to measure the impact of the different cryptographic algorithms on the memory required from a sensing device.

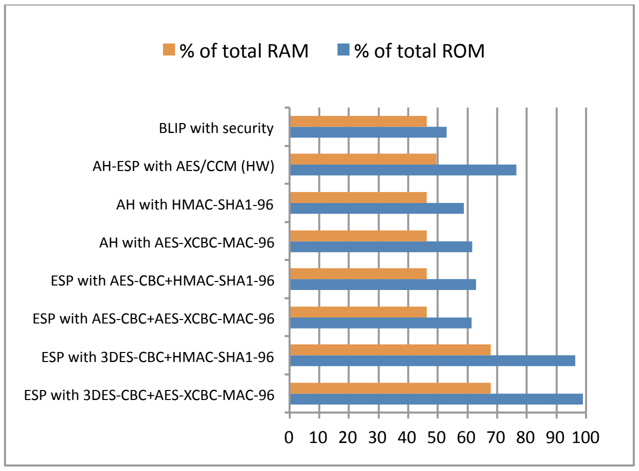

Figure 10. Memory footprint of 6LoWPAN security (percentage of total RAM and ROM memory available in the TelosB)

When compared to the baseline usage profile, we can observe that ESP using cryptographic suites based on 3DES-CBC, both together with HMAC-SHA1-96 and AES-XCBC-MAC-96, is very demanding particularly in terms of the required ROM memory, leaving almost no ROM memory left available to accommodate other mechanisms or applications. The large ROM memory footprint of 3DES-CBC is mostly due to the usage of large S-Boxes by the algorithm. We can also note that the usage of the hardware-level encryption doesn't come without a non-negligible overhead on memory, particularly in terms of ROM memory, as code is necessary to support the usage of link-layer standalone encryption using the cc2420 chip of the TelosB. Our tests were performed using the standalone hardware encryption code available from the Shanghai Jiao Tong University [21]. In contrast to the inline mode, standalone encryption allows applications to perform hardware encryption and decryption without requiring the transmission or reception of a packet by the link-layer, given that such operations are controlled at a higher level in BLIP. From Figure 10 we can also observe that security suites based on the usage of AES-CBC with HMAC-SHA1-96 or AES-XCBC-MAC-96 broadly present a similar impact on the required ROM memory, while requiring only a few more bytes of RAM memory compared to the base 6LoWPAN security application.

From this analysis we are therefore able to conclude that, regarding requirements of memory available in resource-constrained sensing devices, AES appears as a natural candidate in providing an alternative to 3DES-based security suites. AES provides good security both in the CCM and CBC modes with a lower memory footprint. Of course, the usage of AES/CCM on devices that support hardware encryption presents the advantage of freeing more memory for other mechanisms and applications, and in this case AES-CBC can probably be dispensed. Regarding the support of integrity and authentication, AES-XCBC-MAC-96 represents a good choice regarding the required memory, also because it provides superior security to HMAC-SHA1-96 with a similar memory footprint. It is interesting to note that, excluding the cryptographic suites using 3DES-CBC, security in general causes a relatively low overhead in terms of memory. The impact on the available memory of sensing devices therefore doesn't compromise the adoption of network-layer security mechanisms in the context of the 6LoWPAN adaptation layer.

C. *Energy overhead of 6LoWPAN security*

As many sensing applications are designed with battery-powered sensing devices in mind, the energy required from such devices to perform security operations is a critical aspect, given that it influences the expected lifetime of the device and of the overall sensing application. Energy is therefore an important evaluation criterion of the feasibility of any communications or security proposal for smart objects, and one that we evaluate for the usage of 6LoWPAN security.

In Figure 11 we represent the experimentally obtained values of the energy required to process security for a 6LoWPAN packet with 32, 64, 96 or 102 bytes, using the previously identified cryptographic suites. The energy values represented are in millijoules (mJ), and the labels illustrate the energy required for the processing of security in the case of a fully-sized 102 bytes 6LoWPAN packet. Figure 11 allow us to perform a qualitative analysis of the impact of security on the energy available in smart objects, while the values obtained experimentally are later used in the context of our quantitative study on the lifetime of particular sensing applications. Please note that the values represented in Figure 11 already include the energy required for the processing of 6LoWPAN security headers (for its interpretation and construction) in the BLIP networking stack. Also, note that we don't represent the energy required for the processing of a 6LoWPAN packet without any cryptographic operation, because such value is negligible when compared to the energy required for security. The values represented are considered irrespective of the size of the MIC code generated by a specific authentication algorithm. This is due to the fact that AES-XCBC-MAC-96 and HMAC-SHA1-96 always generate 12-byte MIC codes, while for hardware AES/CCM the energy required for the generation of a 16, 12 or 8 bytes MIC using standalone hardware encryption is the same, as in this case hardware security is designed to operate on blocks of 128 bits (16 bytes).

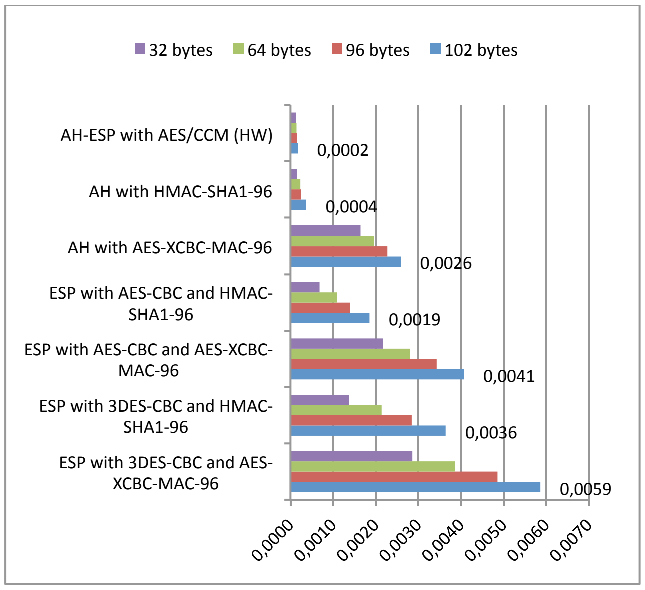

Figure 11. Energy required by 6LoWPAN security (mJ)

From the previous Figure we again observe that cryptographic suites employing 3DES-CBC are clearly less efficient in terms of the energy required, as for example 0.0059mJ are required to encrypt a 102-bytes 6LoWPAN packet and generate the correspondent MIC code using AES-XCBC-MAC-96. Regarding the support of authentication and integrity, we note the notorious difference between HMAC-SHA1-96 and AES-XCBC-MAC-96, which allows us to conclude that the greater security of the later when compared to the former probably does not compensate its impact on energy. In fact, HMAC-SHA1-96 only requires 0.00037 mJ to encrypt a 102 bytes 6LoWPAN packet, while with AES-XCBC-MAC-96 0,0026 mJ are required to process the same packet. From the previous Figure we can also confirm that standalone hardware encryption using the cc2420 chip of the TelosB is extremely energy-efficient, and should therefore provide a superior solution to support integrity, authentication and encryption for 6LoWPAN security in devices where hardware security is available. As expected, encryption using AES/CCM is also clearly superior to AES-CBC implemented in software. It is also interesting to note the superior performance of HMAC-SHA1-96 even when implemented in software, as in reality it appears as not much expensive than hardware AES/CCM.

D. *Computational overhead of 6LoWPAN security*

Other than the memory and energy required to process 6LoWPAN security, the computational effort required from smart objects for security operations is also a relevant aspect. As advanced mechanisms such as multi-threading are usually not supported in low-end microcontrollers such as the MSP430 of the TelosB, the computational time required to process security for a 6LoWPAN packet directly influences the maximum communications rate that a smart object can expect to achieve for a given sensing application.

In Figure 12 we illustrate the computational time required for the processing of a 6LoWPAN packet of different sizes, considering the cryptographic suites previously identified. The values illustrated are in milliseconds (ms) and, as for our previous analysis, we do not represent the computational time required for the processing of 6LoWPAN security without any cryptographic operations, given that such value is negligible when compared to the effort required to process security for the same 6LoWPAN packet. The labels indicate the computational time required to process security for a 102 bytes 6LoWPAN packet. Also note that the values illustrated in Figure 12 are total values, measured from the reception of a 6LoWPAN packet to the time when the respective cryptographic algorithm finishes processing the packet, and therefore represents the total computational effort required to process security for a 6LoWPAN packet in the TelosB. The values were obtained from measurements using the 32 KHz internal oscillator of the TelosB, which is accessible to TinyOS applications via the counter interface.

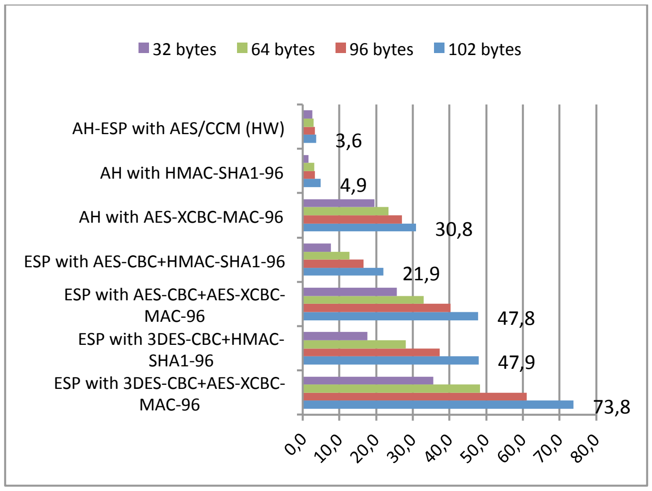

Figure 12. Computational time required by 6LoWPAN security (ms)

From Figure 12 we observe that the most demanding cryptographic suite appropriate to ESP is 3DES-CBC when used together with AES-XCBC-MAC-96, requiring in total approximately 74 ms for processing a 102 bytes 6LoWPAN packet. Regarding the support of integrity and authentication, HMAC-SHA1-96 appears as the most efficient algorithm available in software. AES-XCBC-MAC-96, although providing greater security is much mode demanding, requiring approximately 31 ms to process a fully-sized 6LoWPAN packet. Standalone hardware encryption appears again as the most efficient solution, and in this case the time required to process the same 6LoPWAN packet was measured as 3.6 ms. As AES/CCM implements the CCM* combined mode, this in reality represents the time necessary to encrypt, decrypt or generate the authentication data for a 6LoWPAN packet. Regarding AES implemented in software, we observe that AES-CBC is clearly more demanding, although better than 3DES-CBC in providing confidentiality.

VI. OVERALL EVALUATION OF 6LOWPAN SECURITY

Our experimental evaluation study on the resources required from constrained sensing devices to support 6LoWPAN security allows us to consider its impact in more concrete application scenarios. We therefore proceed to discuss the viability of our proposal regarding sensing applications with diverse requirements in terms of security, communication rates and lifetime of sensing devices.

A. *Impact of 6LoWPAN security on the communications rate of sensing devices*

As sensing applications may be very diverse in terms of the employed communications rate, we find it appropriate to evaluate if 6LoWPAN security may represent a bottleneck in this respect. This is an important evaluation aspect since, as we have seen, security introduces a non-negligible computational overhead on constrained smart objects, which are unable to process packets received or waiting transmission while the microcontroller is busy performing cryptographic operations. When considering communications using IEEE 802.15.4 at 250Kbit/s, we realize that the impact of the computational time required for security on the maximum transmission rate is much larger than the impact on the time required for the transmission of a few more bytes required for the 6LoWPAN security headers and the MIC code. What we cannot exclude from consideration is the overhead introduced by IEEE 802.15.4 addressing on the bandwidth available for 6LoWPAN, which represents 19.6 % of the total bandwidth, as 25 bytes are required for link-layer information with each 127 bytes 6LoWPAN packet (please refer to Figure 1).

In Figure 13 we illustrate the maximum transmissions rate which can be achieved by sensing application employing 6LoWPAN security, considering the usage of the various cryptographic suites with 6LoWPAN packets with 32, 64, 96 or 102 bytes. The values obtained and illustrated in this Figure are in packets per second and are valid for AH and ESP in both tunnel and transport modes, together with the transmission of the authentication data, if required. The values illustrated in Figure 13 consider the time required for the processing of 6LoPWAN headers (including security) on the TelosB, which we have experimentally measured as 0.09 milliseconds. We do not represent the values for the maximum transmission rate without security, but those values are fundamentally greater, in particular 252 packets per second for 102 bytes 6LoWPAN packets, 268 for 64 byte packets, 402 for 96 bytes packets and 803 for 32 bytes packets. From Figure 13 we can observe that the impact of 6LoWPAN security is particularly relevant when transmitting smaller packets. For larger packets (for example for packets measuring from 64 to 102 bytes) security still allows acceptable transmission rates, particularly using cryptographic suites based on AES/CCM and SHA1. One possible design approach for 6LoWPAN applications would therefore be to employ aggregation of sensing data whenever applicable, as this allows reducing the impact of security on the communications rate.

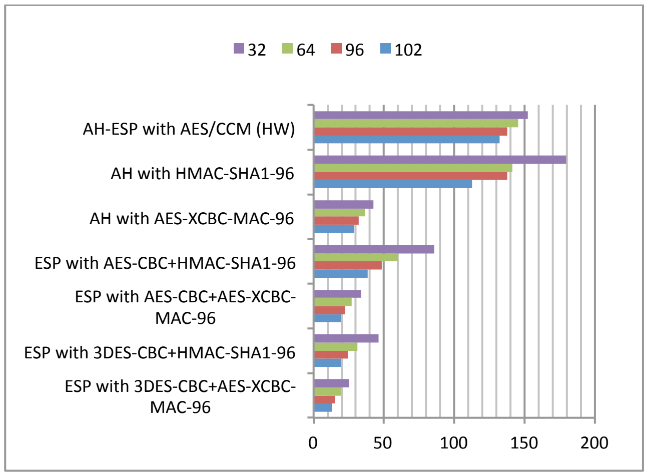

Figure 13. Maximum transmission rate achievable with 6LoWPAN and security (in packets per second)

As visible in Figure 13, security configurations employing 3DES cause a greater impact on the maximum available communications rate. For applications requiring only integrity and authentication, AH using HMAC-SHA1-96 or hardware AES/CCM are good choices. HMAC-SHA1-96 appears in fact again as a superior choice in providing such security properties using a software implemented security algorithm. Again regarding authentication and integrity, HMAC-XCBC-MAC-96 causes a greater impact as can be observed in the previous Figure. It can be nevertheless an appropriate choice for applications requiring lower transmission rates, as it provides security superior to HMAC-SHA1-96. In general, we observe that acceptable transmission rates are achievable using 6loWPAN security. As applications are usually designed in order to save energy by not requiring large transmission rates, the limits identified in Figure 13 should not represent a limitative factor of the applicability of 6LoWPAN security.

B. Impact of 6LoWPAN security on the lifetime of sensing applications

Other than the impact of 6LoWPAN security on the communication rate smart objects are able to achieve, it is also important to analyze its impact on the lifetime of such sensing devices, as it in the end may determine the lifetime of a given sensing application. The importance of this evaluation is related to the fact that most sensing applications designed for the IoT will only be viable if able to operate in unattended mode during a long period of time, as in many situations smart objects are devices for which it is difficult or impossible to replace batteries during long periods of time. As for our previous evaluation studies, our overall goal is to analyze if acceptable compromises can be achieved between the usage of resources on smart objects and security. In Figures 14 to 17 we illustrate the lifetime that a TelosB sensing device can achieve using 6LoWPAN security to process packets with different sizes and using different communications rates. In particular, we consider the usage of lower transmission rates (from 1 to 10 transmitted packets per second) and higher transmission rates (from 20 to 200 transmitted packets per second). We also consider the processing of 32 and 102 bytes 6LoWPAN packets, as this represents two complementary scenarios in terms of the size of 6LoWPAN packets processed in such communications. The Figures illustrate the achievable lifetime in days for each security and usage configurations, and due to the wide range of values we use a logarithmic scale for the representation of the obtained values.

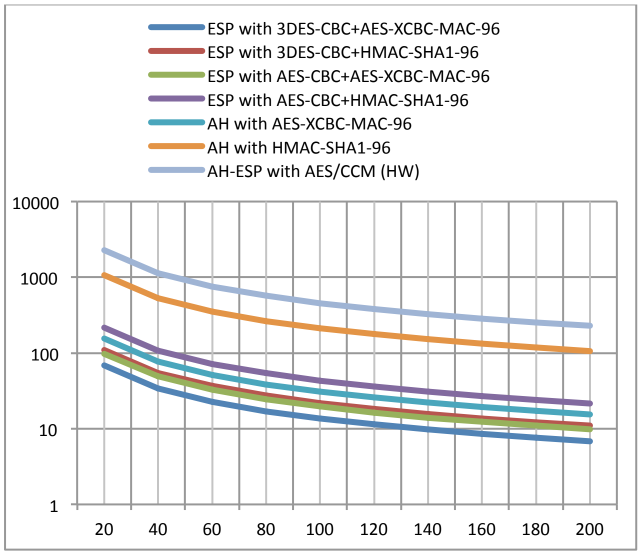

Figure 14. Lifetime of a sensing device in days, when processing security for a 102 bytes 6LoWPAN packet, considering higher communication rates (from 20 to 200 packets/sec).

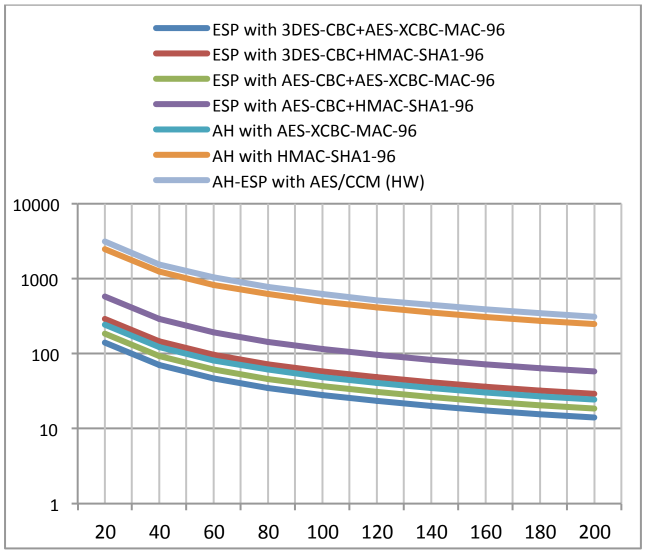

Figure 15. Lifetime of a sensing device in days, when processing security for a 32 bytes 6LoWPAN packet, considering higher communication rates (from 20 to 200 packets/sec).

The values illustrated in Figures 14 to 17 are derived from our experimentally obtained values using a TelosB mote powered using two new AA LR6-type batteries. As for our previous evaluation, we also consider the energy required for the processing of 6LoWPAN headers in each packet (including security headers), which was experimentally measured as 0.007 nanojoules (nJ) per 6LoWPAN processed packet with security. This value reflects the total energy required for the processing of a 6LoWPAN packet, from the invocation of the transmission of the packet using the BLIP networking stack to the time of the completion of its transmission.

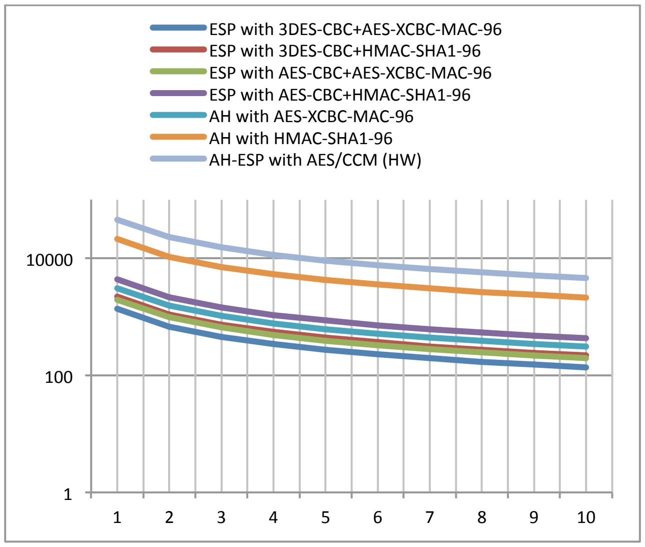

Figure 16. Lifetime of a sensing device in days, when processing security for a 102 bytes 6LoWPAN packet, considering lower communication rates (from 1 to 10 packets/sec).

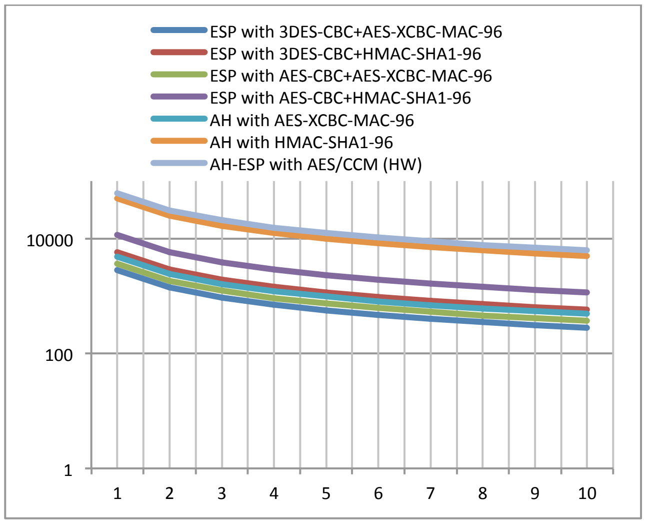

Figure 17. Lifetime of a sensing device in days, when processing security for a 32 bytes 6LoWPAN packet, considering lower communication rates (from 1 to 10 packets/sec).

As with our study on the impact of security on the transmissions rate, we do not consider the extra energy required for the transmission of 6LoWPAN headers in tunnel mode versus transport mode, neither for the transmission of authentication data. This is due to the fact that the energetic cost of the transmission or reception per bit with the TelosB is very small, and consequently represents a negligible impact on the lifetime of the applications and doesn't influence our analysis and conclusions.

From Figures 14 to 17 we observe that AES-CCM and HMAC-SHA1-96 for integrity and authentication allow much higher lifetime of the sensing device, particularly for applications requiring lower transmission rates. 3DES-CBC appears again as a bad choice independently of the transmission rate, while cryptographic suites employing AES-CBC and XCBC-MAC-96 in software are possible choices if applications requiring lower transmission rates. For the processing and transmission of smaller (32 bytes) 6LoWPAN packets, the impact of authentication and integrity using AH with HMAC-SHA1-96 is almost equal to hardware AES/CCM. HMAC-SHA1-96 can therefore be a good alternative in providing authentication and integrity for applications transmitting smaller data payloads, in particular for the usage with smart objects that do not support hardware encryption. It can also be observed that the achievable lifetime using 6LoWPAN security is in general very good, in particular for sensing applications that require lower transmission rates. As many applications on the IoT will probably employ lower transmission rates, we can see that in such situation the other cryptographic suites based on the usage of software AES-CBC and of AES-XCBC-MAC-96 are also viable. It is therefore perfectly possible to employ such cryptographic suites both in software and hardware (for smart objects supporting it) with 6LoWPAN while not critically impacting the lifetime of the sensing device. This factor, together with the conclusions obtained in our previous evaluation studies, allows us to enforce our conviction on the effectiveness of the usage of 6LoWPAN security in the context of a security architecture appropriately designed for the IoT.

VII. CONCLUSIONS

The IPv6 protocol and the 6LoWPAN adaptation layer can play a major role in the evolution of the Internet as we know it today towards the IoT. As the Internet embraces sensorial capabilities, new and exciting applications may come to life that will require and beneficiate from the availability of end-to-end network-layer communications between smart objects and other sensing devices or Internet hosts. Such communications can only be viably employed in the Internet if appropriate security mechanisms are adopted, namely in the context of the 6LoWPAN adaptation layer.

In the current paper we proposed and experimentally evaluated new compressed security headers for 6LoWPAN, and such headers were designed in a way such as to ease its integration with the Internet Security architecture as it evolves in the future. As we have discussed throughout the paper, 6LoWPAN security can be viably employed in various configurations by sensing applications with different requirements in terms of communications rates and payload space. We show that network-layer security can be a reality for sensorial applications used in the context of the IoT. As the proposed mechanisms allow for the usage of different security configurations, security can be adapted to the particular requisites of each sensing application, therefore allowing the establishment of acceptable compromises between security and the usage of resources on limited sensing devices.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the EU Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 224282, Project GINSENG. The authors also thank the Communications and Telematics Group of the CISUC-University of Coimbra.

REFERENCES

[1] IEEE std. 802.15.4 – 2006: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). IEEE, 2006.

[2] N. Kushalnagar et al., “RFC 4919 - IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals,” Aug. 2007; http://tools.ietf.org/html/rfc4919

[3] G. Montenegro et al., “RFC 4944 - Transmission of IPv6 Packets over IEEE 802.15.4 Networks,” Sep. 2007; http://tools.ietf.org/html/rfc4944
G. Montenegro et al., “RFC 4944 - Transmission of IPv6 Packets over IEEE 802.15.4 Networks,” Sep. 2007; http://tools.ietf.org/html/rfc4944

[4] Park, S., Kim, K., Haddad, W., Chakrabarti, S. and Laganier, J. “IPv6 over Low Power WPAN Security Analysis”, Internet Draft, Mar. 2011

[5] Granjal, J.; Silva, R.; Monteiro, E.; Sa Silva, J.; Boavida, F.; , "Why is IPSec a viable option for wireless sensor networks," *Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008. 5th IEEE International Conference on* , vol., no., pp.802-807, Sept. 29 2008-Oct. 2 2008

[6] Granjal, J.; Monteiro, E.; Sá Silva, J.; , "A secure interconnection model for IPv6 enabled wireless sensor networks," *Wireless Days (WD), 2010 IFIP* , vol., no., pp.1-6, 20-22 Oct. 2010

[7] Granjal, J.; Monteiro, E.; Sá Silva, J.; , "Enabling Network-Layer Security on IPv6 Wireless Sensor Networks," *GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference* , vol., no., pp.1-6, 6-10 Dec. 2010

[8] “TinyOS Operating System”; http://www.tinyos.net/.

[9] Jung et al., “SSL-based Lightweight Security of IP-based Wireless Sensor Networks”, International Conference on Advanced Information Networking and Applications Workshop

[10] Gupta, V. et al (2005), “Sizzle: A standards-based end-to-end security architecture for the embedded Internet”, Third IEEE International Conference on Pervasive Computing for the Embedded Internet

[11] Casado, L. and Tsigas, P. (2009), “ContikiSec: A Secure Network Layer for Wireless Sensor Networks under the Contiki Operating System”

[12] S. Kent, “RFC 4303 – IP Encapsulating Security Payload,” Dec. 2005; http://tools.ietf.org/html/rfc4303

[13] S. Kent, “RFC 4302 – IP Authentication Header,” Dec. 2005; http://tools.ietf.org/html/rfc4302

[14] S. Kent and K. Seo, “RFC 4301 – Security Architecture for the Internet Protocol,” Dec. 2005; http://tools.ietf.org/html/rfc4301

[15] J. Hui and P. Thubert, “Compression Format for IPv6 Datagrams in Low Power and Lossy Networks (6LoWPAN)”, Internet Draft, Feb. 2011

[16] R. Housley, “RFC 4309 – Using Advanced Encryption Standard CCM Mode with IPsec Encapsulating Security Payload (ESP),” Dec. 2005; http://tools.ietf.org/html/rfc4309

[17] D. Eastlake, “RFC 4305 – Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH),” Dec. 2005; http://tools.ietf.org/html/rfc4305

[18] C. Madson and R. Glenn, “RFC 2404 – The use of HMAC-SHA-1-96 within ESP and AH,” Nov. 1998; http://tools.ietf.org/html/rfc2404

[19] S. Frankel and H. Herbert, “RFC 3566 – The AES-XCBC-MAC-96 Algorithm and Its Use With IPSec,” Sep. 2003; http://tools.ietf.org/html/rfc3566

[20] “TelosB Mote Platform”; http://www.xbow.com/pdf/Telos_PR.pdf.

[21] “Standalone hardware AES Encryption using CC2420”; http://cis.sjtu.edu.cn/index.php/The_Standalone_AES_Encryption_of_CC2420_(TinyOS_2.10_and_MICAz).

[22] Shahid Raza, Simon Duquennoy, Tony Chung, Dogan Yazar, Thiemo Voigt, Utz Roedig. *Securing Communication in 6LoWPAN with Compressed IPsec*. In: 7th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '11), 27-29 June 2011, Barcelona, Spain.