6lo
Internet-Draft
Intended status: Standards Track
Expires: July 12, 2015

D. Migault, Ed.
Orange
T. Guggemos, Ed.
LMU Munich

January 08, 2015

Requirements for Diet-ESP the IPsec/ESP protocol for IoT
draft-mglt-6lo-diet-esp-requirements-00.txt

Abstract

IPsec/ESP is used to secure end-to-end communications. This document lists the requirements Diet-ESP should meet to design IPsec/ESP for IoT.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 12, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Requirements notation 2
2. Introduction 2
3. Terminology 3
4. Protocol Design 3
5. Byte-Alignment 3
6. Crypto-Suites 4
7. Compression 4
8. Flexibility 5
9. Code Complexity 5
10. Usability 5
11. Compatibility with IP compression Protocols 6
12. Compatibility with Standard ESP 6
13. IANA Considerations 6
14. Security Considerations 6
15. Acknowledgment 6
16. Normative References 7
Appendix A. Power Consumption Example 7
Appendix B. Document Change Log 8
Authors' Addresses 8

1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

IoT devices can carry all kind of small applications and some of them require a secure communication. They can be life critical devices (like a fire alarm), security critical devices (like home theft alarms) and home automation devices. Smart grid is one application where supplied electricity is based on information provided by each home. Similarly, home temperature might be determined by servo-controls based on information provided by temperature sensors.

Using IPsec [RFC4301] in the IoT world provides some advantages, such as:

- IPsec secures application communications transparently as security is handled at the IP layer. As such, applications do not need to be modified to be secured.

- IPsec does not depend on the transport layer. As a result, the security framework remains the same for all transport protocols, like UDP or TCP.

- IPsec is well designed for sleeping nodes as there are no sessions.

- IPsec defines security rules for the whole device, which outsource the device security to a designated area. Therefore IPsec can be seen like a tiny firewall securing all communication for an IoT device.

IPsec is mostly implemented in the kernel, whereas application are in the user space. This is often considered as a disadvantage for IPsec. However, as there are no real distinctions between these two spaces in IoT and that IoT devices are mostly designed to a specific and unique task, this may not be an issue anymore.

IoT constraints have not been considered in the early design of IPsec. In fact IPsec has mainly been designed to secure infrastructure. This document describes the requirements of Diet-ESP, the declination of IPsec/ESP for IoT, enabling optimized IPsec/ESP for the IoT.

3. Terminology

- IoT: Internet of Things

4. Protocol Design

Diet-ESP is based on IPsec/ESP and is adapted for IoT. Adaptation to IoT scenarios must not be at the expense of security, and the security evaluation of Diet-ESP should benefit as far as possible from the long experience of already existing protocols. As a result the protocol design requirements for Diet-ESP are as follows:

R1: Diet-ESP MUST benefit from the IPsec/ESP security.

R2: Diet-ESP MUST NOT introduce vulnerabilities over IPsec/ESP. This means that at some points IPsec/ESP is implemented. A foreseen way to reach that goal is to associate IPsec/ESP with compressors/decompressors.

R3: Diet-ESP SHOULD rely on existing protocol or frameworks.

5. Byte-Alignment

IP extension headers MUST have 32 bit Byte-Alignment in IPv4 (section 3.1 of [RFC0791] - Padding description) and a 64 bit Byte-Alignment in IPv6 (section 4 of [RFC2460]). As ESP [RFC4303] is such an extension header, padding is mandatory to meet the alignment constraint. This alignment is mostly caused by compiler and OS

requirements dealing with a 32 or 64 Bit processor. In the world of IoT, processors and compilers are highly specialized and alignment is often not necessary 32 Bit, but 16 or 8 bit. As a result, the byte-alignment requirement is as follows:

R4: Diet-ESP MUST support Byte-Alignment that are different from 32 bits or 64 bits to prevent unnecessary padding.

R5: Each peer MUST be able to advertise and negotiate the Byte-Alignment, used for Diet-ESP. This could be done for example during the IKEv2 exchange.

6. Crypto-Suites

IEEE 802.15.4 defines AES-CCM*, that is AES-CTR and CBC-MAC, for link layer security with upper layer key-management. Therefore it is usually supported by hardware acceleration. This leads to the following crypto-suite requirement:

R6: Diet-ESP MUST support AES-CCM and MUST be able to take advantage of AES-CCM hardware acceleration. Diet-ESP MAY support other modes.

7. Compression

Sending data is very expensive regarding to power consumption, as illustrated in Appendix A. Compression can be performed at different layers. An encrypted ESP packet is an ESP Clear Text Data encrypted and eventually concatenated with the Initialization Vector IV to form an Encrypted Data Payload. This encrypted Data Payload is then placed between an ESP Header and an ESP Trailer. Eventually, this packet is authenticated with an ICV appended to ESP Trailer. Compression can be performed at the ESP layer that is to say for the fields of the ESP Header, ESP Trailer and the ICV. In addition, ESP Clear Text Data may also be compressed with non ESP mechanisms like ROHC [RFC3095], [RFC5225] for example, resulting in a smaller payload to be encrypted. If ESP is using encryption, these mechanisms MUST be performed over the ESP Clear Text Data before the ESP/Diet-ESP processing as missing of encrypted fields make decryption harder. As a result, compression requirements are as follows:

R7: Diet-ESP MUST be able to compress/remove all static ESP fields (SPI, Next Header) as well as the other fields SN, Padding, Pad Length or ICV.

R8: Diet-ESP SHOULD also allow compression mechanisms before the IPsec/ESP processing.

R9: Diet-ESP SHOULD NOT allow compressed fields, not aligned to 1 byte in order to prevent alignment complexity. In other words, Diet-ESP do not consider finer granularity than the byte.

8. Flexibility

Diet-ESP can compress some of the ESP fields as Diet-ESP is optimized for IoT. Which field may be compressed or not, depends on the scenario and current and future scenarios cannot been foreseen. As a result, the flexibility requirements are as follows:

R10: Diet-ESP MUST be able to compress any field independently from another.

R11: Diet-ESP SHOULD provide different ways to compress a single field, so the most appropriated way can be agreed between the peers.

R12: Each peer MUST be able to announce and negotiate the different compressed fields as well as the used method.

In fact Diet-ESP and ESP differs in the following point: ESP has been designed so that any ESP secured communication so any device is able to communicate with another. This means that ESP has been designed to work for large Security Gateway under thousands of connections, as well as devices with a single ESP communication. Because, ESP has been designed not to introduce any protocol limitations, counters and identifiers may become over-sized in an IoT context.

9. Code Complexity

IoT devices have limited space for memory and storage, which leads to the following requirement.

R13: Diet-ESP MUST be able to be implemented with minimal complexity. More especially, Diet-ESP MUST consider small implementation that implement only a subset of all Diet-ESP capabilities without requiring involving standard ESP, specific compressors and de-compressors.

10. Usability

Application Developer usually do not want to take care about the underlying protocols and security. In addition, the security configuration should remain feasible by a standard software developer. The usability requirements regarding Diet-ESP are as follows:

R14: Diet-ESP MUST remain independent from the application.

R15: Diet-ESP MUST detail for each field how compression impacts the security of the device. Although the creation of profiles is out of scope of Diet-ESP, it is expected that profiles may be defined latter by the usage.

11. Compatibility with IP compression Protocols

There are different protocols providing IP layer compression for constraint devices like IoT (6LoWPAN [RFC6282]) or Mobile Devices (ROHC). The requirements regarding interactions of Diet-ESP and additional compression protocols are as follows:

R16: Diet-ESP MUST be able to interact with IP compression protocols. More especially, this means that a Diet-ESP packet MUST be able to be sent in a ROHC or a 6LowPAN packet. Diet-ESP document should explicitly detail how this can be achieved.

R17: Diet-ESP MUST also detail how compression of layers above IP with ROHC or 6LowPAN is compatible with Diet-ESP.

12. Compatibility with Standard ESP

IPsec/ESP is widely deployed by different vendors on different machines. IoT devices MAY have to communicate with Standard ESP implementations. The ESP compatibility requirements is as follows:

R18: Diet-ESP MUST be able to communicate with Standard ESP.

13. IANA Considerations

There are no IANA consideration for this document.

14. Security Considerations

Security Considerations have been expressed as one of the requirement.

15. Acknowledgment

The current work on Diet-ESP results from exchange and cooperation between Orange, Ludwig-Maximilians-Universitaet Munich, Universite Pierre et Marie Curie. We thank Daniel Palomares and Carsten Bormann for their useful remarks, comments and guidances on the design. We thank Sylvain Killian for implementing an open source Diet-ESP on Contiki and testing it on the FIT IoT-LAB [fit-iot-lab] funded by the French Ministry of Higher Education and Research. We thank the IoT-

Lab Team and the INRIA for maintaining the FIT IoT-LAB platform and for providing feed backs in an efficient way.

16. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2460] Deering, S.E. and R.M. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC3095] Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T., and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, July 2001.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC5225] Pelletier, G. and K. Sandlund, "RObust Header Compression Version 2 (ROHCv2): Profiles for RTP, UDP, IP, ESP and UDP-Lite", RFC 5225, April 2008.

[RFC6282] Hui, J. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, September 2011.

[fit-iot-lab] , "Future Internet of Things (FIT) IoT-LAB", , <https://www.iot-lab.info>.

Appendix A. Power Consumption Example

IoT devices are often installed once and left untouched for a couple of years. Furthermore they often do not have a power supply wherefore they have to be fueled by a battery. This battery may have a limited capacity and maybe not replaceable. Therefore, power can be a limited resource in the world of IoT. Table 1 and Table 2 shows the costs for transmitting data and computation

Note these data are mentioned here with an illustrative purpose, for our motivations. These data may vary from one device to another, and may change over time.

	power consumption
low-power radios < 10mW	(100nJ - 1uJ) / bit

Table 1: Power consumption for data transmission.

	power consumption
energy-efficient microprocessors	0.5nJ / instruction
high-performance microprocessors	200nJ / instruction

Table 2: Power consumption for computation.

From these tables, sending 1 bit costs as much as 10-100 instructions in the CPU. Therefore there is a high interest to reduce the number of bits sent on the wire, even if it generates costs for computation.

Appendix B. Document Change Log

[draft-mglt-6lo-diet-esp-requirements-00.txt]: Published: Minor re-wordings

[draft-mglt-ipsecme-diet-ipsec-requirements-00.txt]: First version published.

Authors' Addresses

Daniel Migault (editor)
Orange
38 rue du General Leclerc
92794 Issy-les-Moulineaux Cedex 9
France

Phone: +33 1 45 29 60 52
Email: mglt.ietf@gmail.com

Tobias Guggemos (editor)
LMU Munich
Am Osteroesch 9
87637 Seeg, Bavaria
Germany

Email: tobias.guggemos@gmail.com