IPSECME
Internet-Draft
Intended status: Standards Track
Expires: September 4, 2014

D. Migault (Ed)
Orange
T. Guggemos
Orange / LMU Munich
D. Palomares
Orange / LIP6 - UMPC

March 3, 2014

Diet-ESP: a flexible and compressed format for IPsec/ESP
draft-mglt-ipsecme-diet-esp-00.txt

Abstract

IPsec/ESP has been designed to secure IP packets exchanged between two nodes. IPsec implements security at the IP layer which makes security transparent to the applications, as opposed to TLS or DTLS that requires application to implement TLS/DTLS. As a result, IPsec enable to define the security rules in a similar way one establishes firewall rules.

One of the IPsec's drawbacks is that implementing security on a per packet basis adds overhead to each IP packet. Considering IoT devices, the data transmitted over an IP packet is expected to be rather small, and the cost of sending extra bytes is so high that IPsec/ESP can hardly be used for IoT as it is currently defined in RFC 4303.

This document defines Diet-ESP, a protocol that compress and reduce the ESP overhead of IPsec/ESP so that it can fit security and energy efficient IoT requirements. Diet-ESP use already existing mechanism like IKEv2 to negotiate the compression format. Furthermore a lot of information, already existing for an IPsec Security Association, are reused to offer light negotiation in addition to maximum compression.

Status of This Memo

Table of Contents

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

The IPsec/ESP [RFC4303] is represented in Figure Figure 1 . It was designed to: 1) provide high level of security as a basis, 2) favor interoperability between implementations 3) scale on large infrastructures.

In order to match these goals, ESP format favor mandatory fields with fixed sizes that are designed for the worst case scenarios. This results in a kind of "unique" packet format common to all considered scenarios using ESP. These specific scenarios MAY result in carrying "unnecessary" or "larger then required" fields. This cost of additional bytes were than considered as negligible versus interoperability, and this made ESP very successful over the years.

With IoT, requirements become slightly different. For most devices, like sensors, sending extra bytes directly impacts the battery and so the life time of the sensor. Furthermore, IoT scenarios MAY consider that sensors MAY be designed not to interconnect between each other, but instead to be connected to a specific Security Gateway. These kind of dedicated connectivity, for example, does not impose the sensors to be fully interoperable with any other IPsec/ESP implementation. In contrast, it MAY be inter-operable with the Security Gateway and those devices supporting the same sensor's options.

In this document, we adapted ESP so IoT devices can use ESP designed for their specific needs or applications. Diet-ESP allows to reduce or remove all fields of the ESP format represented in figure 1. How the fields are reduced is defined in the Diet-ESP Context. This Diet-ESP Context MAY be announced or negotiated between the two peers. How the two devices agree on using the same Diet-ESP Context is out of scope of this document. Diet-ESP Context consist of a byte that fully defines the parameters present in a Diet-ESP packet, creating a Diet-ESP packet format agreement between compliant devices.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ----
|               Security Parameters Index (SPI)                 | ^Int.
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|                      Sequence Number                          | |ered
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | ----
|                     Payload Data* (variable)                  | |   ^
~                                                               ~ |   |
|                                                               | |Conf.
+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|               |     Padding (0-255 bytes)                     | |ered*
+-+-+-+-+-+-+-+-+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |   |
|                               |  Pad Length   | Next Header   | v   v
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ------
|         Integrity Check Value-ICV   (variable)                |
~                                                               ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: ESP Packet Description

3. Terminology

This document uses the following terminology:

- IoT: Internet of Things

- ESP: ESP like described in [RFC4303].

- ESP-packet: The concatenation of the following fields:

  - ESP-Header: The concatenation of the SPI and SN

  - ESP Payload: The concatenation of the following two fields. The ESP Payload is usually encrypted.

    - Data Payload: The application payload. It MAY include a transport layer header.

    - ESP-Trailer: The Padding concatenated with the Pad Length and Next Header fields.

  - ESP ICV The ICV generated throw the specified algorithm.

- Diet-ESP: Diet version of ESP like described in this document.

- Diet-ESP Context: The Context that describes the Diet-ESP packet format (see Section 5).

- Diet-ESP-packet: The concatenation of the following fields:

    - Diet-ESP-Header: The concatenation of the SPI and SN if they appear in the packet.

    - Diet-ESP Payload: The concatenation of the following two fields. The Diet-ESP Payload is usually encrypted.

        - Data Payload: The application payload. If the transport layer header is present, it MAY be removed.

        - Diet-ESP-Trailer: The Padding concatenated with the Pad Length and Next Header fields if they appear in the packet.

    - Diet-ESP ICV: The ICV generated throw the specified algorithm and MAYBE truncated by Diet-ESP.

4. Diet-ESP: Protocol Description

This section describes how each field of the ESP can be compressed.

SPI SIZE: ESP Security Policy Index is 32 bits long. Diet-ESP omits, leaves unchanged, or reduces the SPI to 8, 16 or 24 bits. The length of the SPI should be guided by 1) the number of simultaneous inbound SA the device is expected to handle and 2) reliability of the IP addresses in order to identify the proper SA for incoming packets. More specifically, a sensor with a single connection to a Security Gateway, may bind incoming packets to the proper SA based only in its IP addresses. In that case, the SPI MAY not be necessary. Other scenarios may consider using the SPI to index the SAs or may consider having multiple ESP channels with the same host from a single host. In that case it may choose a reduced length for the SPI. Note that reducing the size of the SPI may expose the system to security flows. See Section 8 for more details. Note also that the value 0 for the SPI is note allowed to be sent on the wire as described in [RFC4303].

For those cases where a regular SPI of 32 bits has been negotiated (e.g. via IKEv2 [RFC5996]), the resulting SPI used for Diet-ESP packets corresponds to the high order bits of that 32 bits SPI (see Section 6 for further explanations).

SN SIZE: ESP Sequence Number is 32 bit and extended SN is 64 bit long. Diet-ESP omits, leaves unchanged or reduces SN to 8, 16, 24

bits. The length of the SN should be guided by 1) how the receiving side handles the SN, 2) the number of packets expected to be sent over Diet-ESP channel, and 3) how the node is willing to use IKEv2 to re-key when SN are expired. SN are used to address replay attacks, thus removing SN may expose the system to security flaws. See Section 8 for more details. If SN is used, a 32 bits value may not be required. Table 1 shows the lifetime of one SA before re-keying is required in case the SN expires.

SN Length	1 packet per second	1 packet per minute	1 packet per hour
8 bit	4min 16sec	4h 16min	10 days 16h
16 bit	18h 12min 16sec	6 weeks 3 days 12h	~7 years 25 weeks
24 bit	~27 weeks 5 days	31 years 47 weeks	~1,915 years
32 bit	~136 years	~8,171 years	~490,293 years

Table 1: Lifetime of one Security Association with different sizes of Sequence Numbers compared to different use cases.

Note that SN and SPI MUST be aligned to a multiple of the Alignment value (ALIGN).

NH: Diet-ESP is able to remove the Next Header field from the ESP-Trailer if the underlying protocol can be derived from the Traffic Selector (TS) within the SA. More specifically, the next header indicates whether the encrypted ESP payload is an IP packet, a UDP packet, a TCP packet or no next header. The NH can only be removed if this has been explicitly specified in the SA or if the device has a single application. Suppose a device sets an ESP channel with another peer only considering the IP addresses as TS without specifying the transport protocols or (or upper layer protocols). If the device uses this channel for multiple upper layer protocols (like HTTP and tnftp), then the NH cannot be removed as the receiver would not be able to determine whether incoming packets are HTTP or tnftp.

Note that removing the Next Header impacts how encryption is performed. For example, the use of AES-CBC [RFC3602] mode requires the last block to be padded to reach a 128 bit alignment. In this case removing the Next Header increases the padding by the Next Header length, which is 8 bits. In this case, removing the Next Header provides few advantages, as it does not reduce the ESP packet length. With AES-CBC, the only advantage of removing the Next Header would be for data with the last block of 15 bytes. In that case, ESP pad with 15 modulo 16 bytes, set the 1 byte pad length field to 15

and add the one byte Next Header field. This leads to 15 + 15 + 1 + 1 bytes to be sent. On the other hand, removing the Next Header would require only the concatenation of the pad length byte with a 0 value, which leads to 16 bytes to be sent.

Other modes like AES-CTR [RFC3686] do not have block alignment requirements. Using AES-CTR with ESP only requires the 32 bit alignment - mostly for OS implementation. In fact if an n byte alignment is required (for encryption or for packet format), data of length k * n + n - 1 bytes, k an integer, takes advantage of removing the Next Header and reduces the data to be sent over n bytes. In the case of sensor network it is very likely that data of fixed size k * n + n - 1 will be used. Furthermore, if 32 bits alignment is reduced to 8 bits alignment, Next Header is always an additional unnecessary byte being sent.

PAD: With ESP, all packets have a Pad Length field. This field is usually present because ESP requires a 32 bits alignment which is performed with padding. Diet-ESP considers that some devices may use 8 bits alignment, in which case padding is not necessary. Similarly, sensors may send application data that has fixed length matching the alignment. Note that alignment may be required by the device (8-bit, 16-bit, or more generally 32-bit), but it may also be required by the encryption block size (AES-CBC uses 128 bit blocks). With ESP these scenarios would result in an unnecessary Pad Length field always set to zero. Diet-ESP considers those case with no padding, and thus the Pad Length field can be omitted.

ALIGN: Alignment for Padding and Pad Length. ESP is designed for 32 bit alignment. This is mostly an OS implementation and hardware design requirements for regular PC processors. IoT may not have these requirements. Having no alignment requirements or a 16 bits alignment requirement prevents or reduces the number of padding bytes to be sent. As a result Diet-ESP considers alternative alignment (8-bit, 16 bit, 32 bit) so to reduce the number of padding bytes.

Note that when PAD requires the Pad Length field to be present, ALIGN provides the minimum alignment padding considers. More specifically, ALIGN gives more priority to the hardware or OS implementation than to the encryption algorithm used. In fact with AES-CTR padding will be performed based on the value provided by ALIGN. However, AES-CBC padding is performed on the AES block basis (128 bits). This value overwrites the one provided by ALIGN.

IH: With ESP using the tunnel mode, the inner IP Header is sent in every ESP Payload. This extra bytes sent do not carry relevant information over sent packets. As a result Diet-ESP indicates the IP

header has been omitted, and MUST be rebuilt by the receiver. These information are negotiated via IKE and are stored in the SA.

TH: With ESP the transport header is transmitted in every packet. This layer may not provide relevant information, especially for UDP transport layer. The port parameters may be negotiated via IKE and stored in the SA. As a result Diet-ESP indicates that the transport protocol header (TH) has been removed from the encrypted ESP Payload. This option can only be used if the header can be restored or if it is unnecessary for the further packet procession. Other protocols than UDP are considered out of scope of this document. TCP, for example, includes information that are not as easy to restore, like options, controls or windows. In order to use other transport layer protocols within specific configuration, additional information may be provided in the future.

ICV: ESP negotiates Authentication protocols. These protocols generate an ICV of a length defined by the authentication protocol negotiated for the SA. These authentication protocols do not provide ways to perform weak authentication, as it only reduces the size of the ICV. IoT is interested in weak authentication as it may sends a small amount of bytes, and the trade-of between battery life time and security may be worth. As a result Diet-ESP indicates the number of bytes of the ICV. Diet-ESP considers sending the whole ICV or the first 1 byte resp (2, 4, 8, 12, 16, 32) bytes. Note that Note that reducing the size of the SPI may expose the system to security flows. See Section 8 for more details.

5. Diet-ESP Context: Format Description

This section describes the Diet-ESP Context that contains all necessary parameters for Diet-ESP.

```
 0                             1
 0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|SPI SIZE|SN SIZE |NH|P |ALIGN|TH|IH|  ICV   | X|
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

Figure 2: Diet-ESP Context

With the fields defined as below:

- SPI SIZE (3 bits): specifies the size of the SPI field length of
      the Diet-ESP header in byte. Values can be from 0 to 4. A
      zero value means the SPI does not appear in the Diet-ESP
      packet. The size depends on the use case, the connection
      should be used for.

  - 000: indicates a 0 bit SPI. The SPI is removed from the packet.

  - 001: indicates an 8 bit SPI in each Diet-ESP-packet.

  - 010: indicates a 16 bit SPI in each Diet-ESP-packet.

  - 011: indicates a 24 bit SPI in each Diet-ESP-packet.

  - 100: indicates a 32 bit SPI in each Diet-ESP-packet. This configuration is according to the RFC 4303 [RFC4303]

  - 101: Unassigned

  - 110: Unassigned

  - 111: Unassigned

- SN SIZE (3 bits): specifies the size of the Sequence Number field within the Diet-ESP header in byte. Values can be from 0 to 4. A zero value means the SN does not appear in the Diet-ESP packet. The size depends on the use case, the connection should be used for.

  - 000: indicates a 0 bit SN. The SN is removed from the packet and anti-replay is disabled on the receiver.

  - 001: indicates an 8 bit SN in each Diet-ESP-packet.

  - 010: indicates a 16 bit SN in each Diet-ESP-packet.

  - 011: indicates a 24 bit SN in each Diet-ESP-packet.

  - 100: indicates a 32 bit SN in each Diet-ESP-packet. This configuration is according to the RFC 4303 [RFC4303]

  - 101: Unassigned

  - 110: Unassigned

  - 111: Unassigned

- NH (1 bit): specifies if the Next Header field appears in the Diet-ESP trailer. NH unset to 0 indicates the Next Header field is present and NH set to 1 indicates the Next Header is omitted.

- P (1 bit): specifies if the Pad Length field appears in the Diet-ESP trailer. P unset to 0 indicates the Pad Length field is present and P set to 1 indicates the Pad Length is omitted.

- ALIGN (2 bits): specifies Padding, Padding Length as follows:

  - 00: indicates an 8 bit alignment. The field Pad Length is omitted and the Diet-ESP packet never has Padding.

  - 01: indicates a 16 bit alignment. The field Pad Length is always present.

  - 10: indicates a 32 bit alignment. The field Pad Length is always present.

  - 11: Unassigned

- TH (1 bit): specifies if the transport layer field appears in the Diet-ESP Payload Data. TH unset to 0 indicates the Transport header field is present and TH set to 1 indicates the transport header is omitted. In this case, the transport protocol MUST be specified in the SA with its associated port. If a non unique port or a non unique transport protocol is specified, this bit MUST be unset to 0. Otherwise, the device will not be able to rebuilt the transport header. This document only considers UDP.

- IH (1 bit): specifies if the inner IP address field appears in the Diet-ESP Payload Data. This bit is only significant for the tunnel mode. With IPsec transport mode, IH SHOULD be set to 0 and ignored. With tunnel mode IH unset to 0 indicates the inner IP header field is present and IH set to 1 indicates the inner IP header is omitted.

- ICV (2 bits): specifies the transmitted number of bytes to authenticate the Diet-ESP packet. Note that ICV is optional so if one chose not to perform authentication, it SHOULD negotiate the authentication algorithm to NULL as defined in [RFC4835]. The minimum length greater than 0 for ICV is 96 bits and can be generated with the following hash functions: HMAC-MD5-96 [RFC2403], HMAC-SHA1-96 [RFC2404], AES-CMAC-96 [RFC4494], AES-XCBC-MAC-96 [RFC3566]. As a result ICV only specifies size lower than 96 bits.

  - 000: ICV is left untouched as it is specified by the authentication algorithm.

- 001: Diet-ESP ICV consists of the 8 most significant bits of ESP ICV.

- 010: Diet-ESP ICV consists of the 16 most significant bits of ESP ICV.

- 011: Diet-ESP ICV consists of the 32 most significant bits of ESP ICV.

- 100: Diet-ESP ICV consists of the 64 most significant bits of ESP ICV.

- 101: Unassigned

- 110: Unassigned

- 111: Unassigned

- X (1 bit): Extension bit. When set to 1, this bit indicates an additional byte carry information. In this document, this bit MUST be set to 0.

6. Difference between Diet-ESP and ESP

This section details how to use Diet-ESP to send and receive messages. The use of Diet-ESP is based on the IPsec architecture [RFC4301] and ESP [RFC4303]. We suppose the reader is familiar with these documents and list here the adaptation that MAY be involved by Diet-ESP.

6.1. Packet Alignment

In ESP each packet has a fixed alignment to 32 bits. For Diet-ESP each device has an internal parameter that defines the minimal kernel alignment that is acceptable. ALIGN SHOULD be a the maximum of the peer's minimal alignment.

Diet-ESP Context with SPI SIZE + SN SIZE that is not a multiple of ALIGN MUST be rejected.

6.2. SAD

6.2.1. Storing SPI SIZE SPI in the SAD

For devices using a single SPI SIZE value (e.g. sensors), the SA will be indexed with the SPI as described in ESP. More specifically, SPI is used as the index in the SAD. The only difference is that it has smaller size in the ESP header. If the only supported SPI SIZE is

zero, the lookup has to be performed with the IP address. Some implementations MAY use a specific SPI value in their SAD for unspecified SPIs.

For devices that allow multiple SPI SIZE, like some IoT generic end points or IoT Security Gateways, SAD lookup has to deal with SPI of different sizes. The SPI stored in the SAD MAY be converted from an SPI of any size to a standard 4 bytes SPI. This means that for inbound packets a conversion from SPI SIZE byte SPI to 4 byte SPI is performed before the SAD lookup.

The SPI of the SA may be negotiated using IKEv2 [RFC5996]. Regular IKEv2 implementation negotiate a 4 byte SPI. In order to be able to use regular IKEv2 for Diet-ESP, the following convention is used. The SPI considered in Diet-ESP consists in the SPI SIZE low power bytes of 32 bit SPI negotiated with IKEv2. Only this value should be considered in the SAD. How the SPI SIZE SPI is represented in the SAD is another issue addressed above.

6.2.2. Inbound Security Association Lookup

For devices that are configured with a single SPI SIZE value can process inbound packet as defined in [RFC4301]. As such, no modifications is required by Diet-ESP.

Detecting Inbound Security Association: Identifying the SA for incoming packets is a one of the main reasons the SPI is send in each packet on the wire. For regular ESP (and AH) packets, the Security Association is detected as follows:

1. Search the SAD for a match on {SPI, destination address, source address}. If an SAD entry matches, then process the inbound ESP packet with that matching SAD entry. Otherwise, proceed to step 2.

2. Search the SAD for a match on {SPI, destination address}. If the SAD entry matches, then process the inbound ESP packet with that matching SAD entry. Otherwise, proceed to step 3.

3. Search the SAD for a match on only {SPI} if the receiver has chosen to maintain a single SPI space for AH and ESP, or on {SPI, protocol} otherwise. If an SAD entry matches, then process the inbound ESP packet with that matching SAD entry. Otherwise, discard the packet and log an auditable event.

For device that are dealing with different SPI SIZE SPI, the way inbound packets are handled differs from the [RFC4301]. In fact, when a inbound packet is received, the peer does not know the SPI

SIZE. As a result, it does not know the SPI that applies to the incoming packet. The different values could be the 0 (resp. 1, 2, 3 and 4) first bytes of the IP payload.

Since the size of the SPI is not known for incoming packets, the detection of inbound SAs has to be redefined in a Diet-ESP environment. In order to ensure a detection of a SA the above described regular detection have to be done for each supported SPI size (in most cases 5 times). In most common cases this will return a unique Security Association.

If there is more than one SA matching the lookup, the authentication MUST be performed for all found SAs to detect the SA with the correct key. In case there is no match, the packet MUST be dropped. Of course this can lead into DoS vulnerability as an attacker recognizes an overlap of one or more IP-SPI combinations. Therefore it is highly recommended to avoid different values of the SPI SIZE for one tuple of Source and Destination IP address. Furthermore this recommendation becomes mandatory if NULL authentication is supported. This is easy to implement as long as the sensors are not mobile and do not change their IP address.

The following optimizations MAY be considered for sensor that are not likely to perform mobility or multihoming features provided by MOBIKE [RFC4555] or any change of IP address during the lifetime of the SA.

Optimization 1 - SPI SIZE is mentioned inside the SPI: The SPI SIZE is defined as part of the SPI sent in each packet. Therefore the receiver has to choose the most significant 2 bits of the SPI in the following way in order to recognize the right size for incoming Diet-ESP packets:

00: SPI SIZE of 1 byte is used.

01: SPI SIZE of 2 byte is used.

10: SPI SIZE of 3 byte is used.

11: SPI SIZE of 4 byte is used.

If the the value 0 is chosen for the SPI SIZE this option does not feasible.

Optimization 2 - IP address based lookup: IP addressed based search is one optimization one MAY choose to avoid several SAD lookups. It is based on the IP address and the stored SPI SIZE, which MUST be the same value for each SA of one IP address. Otherwise it can't neither be ensured that an SA is found nor that the correct one is found.

Note that in case of mobile IP the SPI SIZE MUST be updated for all SAs related to the new IP address which may cause in renegotiation. Figure Figure 3 shows this lookup described below.

1. Search most significant SA as follows:

   1.1 Search the first SA for a match on {destination address, source address}. If an SA entry matches, then process to step 2. Otherwise, proceed to step 1.2.

   1.2 Search the first SA for a match on {source address}. If an SA entry matches, then process to step 2. Otherwise, drop the packet.

2. Identify the size of the compressed SPI for the found SA, stored in the Diet-ESP context. Note that all SAs to one IP address MUST have the same value for the SPI SIZE. Then go to step 3.

3. If the SPI SIZE is NOT zero, read the SPI SIZE SPI from the packet and perform a regular SAD lookup as described in [RFC4301]. If the SPI SIZE is zero, the SA from step 1 is unique and can be used.

Note that some implementation MAY collect all SPI matching the IP addresses in step 2 to avoid an additional lookup over the whole SAD. This is implementation dependant.

```
                            +-----------+
                            |   START   |
                            +-----------+
                                  |
                                  V
                         +-----------------+
                         | find 1st SA to  |
                         |   IP address    |------------------+
                         +-----------------+                  |
                                  |                           |
                              ____V____                       V
                      yes    /         \                   +-----+
                     +------/ SPI_SIZE  \           /----->|'---'|
                     |      \   = 0 ?   /          /       | SAD |
                     |       \_________/          /        +     +
                     |            |no            /          '---'
                     |            V             /
                     |   +-----------------+   /
                     |   | find 1st SA to  |--/
                     |   |     SPI +IP     |
                     |   |     address     |
                     |   +-----------------+
                     |            |
                     |            V
                     |   +-----------------+
                     |   | Diet-ESP packet |
                     +-->|    procession   |
                         +-----------------+
```

Figure 3: SAD lookup for incoming packets.

If the sensor is likely to change its IP address, the outcome MAY be a given IP address associated to different SPI SIZE. This case MAY occur if one IP address has been used by a device not anymore online, but the SA has not been removed. The IP has then been provided to another device. In this case the Diet-ESP Context SHOULD NOT be accepted by the Security Gateway when the new Diet-ESP Context is provided to the Security Gateway. At least the Security Gateway can check the previous peer is reachable and then delete the SA before accepting the new SA.

Another case MAY be that a sensor got two interfaces with different IP addresses, negotiates a different SPI SIZE on each interface and then use MOBIKE to move the IPsec channels from one interface to the other. In this case, the Security Gateway SHOULD NOT accept the update, or force a renegotiation of the SPI SIZE for all SAs, basically by re-keying the SAs.

6.2.3. Outgoing Security Association Lookup

Outgoing lookups for the SPI are performed in the same way as it is done in ESP. The Traffic Selector for the packet is searched and the right SA is read from the SA. The SPI used in the packet MUST be reduced to the value stored in SPI SIZE.

6.3. Sequence Number

Sequence number in ESP [RFC4303] can be of 4 bytes or 8 bytes for extended ESP. Diet-ESP introduces different sizes. One way to deal with this is to add a MAX_SN value that stores the maximum value the SN can have. Any new value of the SN will be check against this MAX_SN.

6.4. Outgoing Packet processing

NH, TH, IH, P indicate fields or payloads that are removed from the Diet-ESP packet. How the Diet-ESP packet is generated depends on the Payload Data of lPD bytes, BLCK the block size of the encryption algorithm and the device alignment ALIGN. We note M = MAX(BLCK, ALIGN). Although not normative the resulting Diet-ESP packet should be and explained below. We consider the Diet-ESP Payload as described in Section 3

- 1: if TH is set to 1, then remove the transport layer of the Payload Data.

- 2: if IH is set to 1, and the IPsec mode is tunnel, then remove the inner IP address of the Payload Data.

- 3: if PAD is set to 0 and NH is set to 0: Diet-ESP considers both fields Pad Length and Next Header. The Diet-ESP Payload is the encryption of the following clear text: Payload Data | Padding of Pad Length bytes | Pad Length field | Next Header field. The Pad Length value is such that lPD + 2 + Pad Length = 0 [M].

- 4: if PAD is set to 0 and NH is set to 1: Diet-ESP considers the Pad Length field but removes the Next Header field. The ESP Payload is the encryption of the following clear text: Payload Data | Padding of Pad Length bytes | Pad Length field | Next Header field. The Pad Length value is such that lPD + 1 + Pad Length = 0 [M].

- 5: if PAD is set to 1 and NH is set to 0: Diet-ESP considers the Next Header but do not consider the Pad Length field or the Padding Field. This is valid as long as lPD + 1 = 0 [M]. If M = 1 as it is the case for AES-CTR this equation is always true.

On the other hand the use of specific block size requires the application to send specific length of application data.

- 6: if PAD is set to 1 and NH is set to 1: Diet-ESP does consider neither the Next Header field nor the Pad Length field nor the Padding Field. This is valid as long as lAD = 0 [M]. If M = 1 as it is the case for AES-CTR this equation is always true. On the other hand the use of specific block size requires the application to send specific length of application data.

6.5. Inbound Packet processing

Decryption is for performed the other way around.

After SAD lookup, authenticating and decrypting the Diet-ESP payload the original packet is rebuild as follows:

- 1: if PAD is set to 1 and NH is set to 1: Diet-ESP does consider neither the Next Header field nor the Pad Length field nor the Padding Field. The Next Header field of the IP packet is set to the protocol defined for incoming traffic within the Traffic Selector of the SA. Because there is no Padding it is disregarded.

- 2: if PAD is set to 1 and NH is set to 0: Diet-ESP considers the Next Header but do not consider the Pad Length field or the Padding Field. The Next Header field of the IP packet is set to the value within the Diet-ESP trailer.

- 3: if PAD is set to 0 and NH is set to 1: Diet-ESP considers the Pad Length field but removes the Next Header field. The Next Header field of the IP packet is set to the protocol defined for incoming traffic within the Traffic Selector of the SA. The Pad Length field is read and the Padding is removed from the Data Payload which results the original Data Payload.

- 4: if PAD is set to 0 and NH is set to 0: Diet-ESP considers both fields Pad Length and Next Header. The Next Header field of the IP packet is set to the value within the Diet-ESP trailer. The Pad Length field is read and the Padding is removed from the Data Payload which results the original Data Payload.

- 5: if IH is set to 1, and the IPsec mode is tunnel and the IP header is reconstructed. The source and destination address and the Next Header field are read from the Traffic Selector. The Payload Length is calculated including the size of the transport header, regardless if it is removed with TH or not. All other IP-header values are set to common defaults or have

to be negotiated otherwise which is out of scope of this document.

-6: if TH is set to 1, the Transport layer header is restored with the information in the Security Association. Section 4 describes some differences between the different protocols. In this document we focus on UDP which can be easily restored with the ports inside the Traffic Selector. The Length field can be calculated and the checksum can be left as 0 according to [RFC0768]

7. IANA Considerations

There are no IANA consideration for this document.

8. Security Considerations

This section lists security considerations related to the Diet-ESP protocol.

Small SPI SIZE exposes the device to DoS. For a device, the number of SA is related to the number of SPI. For systems using small SPI SIZE values as index of their database, the number of simultaneous communications is limited by the SPI SIZE. This means that a given device initiating SPI SIZE communications can isolate the system. In order to leverage this vulnerability, one can consider receiving systems that generate 32 bits SPI with a hash function that considers different parameters associated to the reduced SPI. For example, if one use the IP addresses as well as the reduced SPI, the number of SPI becomes SPI SIZE per IP address. This may be sufficient as sensors are not likely to perform multiple communications.

Small size of ICV reduces the authentication strength. For example 8 bits mean that authentication can be spoofed with a probability of 1/256. Standard value considers a length of 96 bit for reliable authentication. If specified, the ICV field is truncated after the given number of bits which, for sure, has to be mentioned while incoming packet procession as well. For removing authentication ESP NULL has to be negotiated, as described in RFC4303.

Removing the SN prevents protection against replay attack.

9. NAT Considerations

This section lists considerations related to the use of Diet-ESP in NAT environments.

Diet-ESP is a protocol designed for the IoT, assuming to work in an IPv6 environment. However, ESP is designed to work in every IP environment whereas Diet-ESP can be delivered in other IP environments like IPv4 as well. This environment MAY cause the need of NAT. In IPsec, UDP encapsulation is used to deal with NAT environments as described in [RFC3948]. Because UDP cannot define the underlying protocol with a Next Header, IKEv2 traffic is distinguished from ESP or AH traffic by sending the Non-ESP Marker (4 bytes ZERO) after the UDP header. As the SPI is considered to never be ZERO this clearly identifies IKEv2 traffic.

In context of Diet-ESP the SPI MAY not be present in the Diet-ESP-header, which MAY corrupt this mechanism in case:

- the 4 byte SN is ZERO

- the SN is absent and the first 4 byte of the encrypted payload are ZERO

- the compressed SN is ZERO AND the remaining bytes of the encrypted payload are ZERO

Therefore an implementer has to define a way to ensure the first 4 bytes are NOT zero. We suggest the negotiated SPI to be at least 1 byte if UDP encapsulation is enabled.

10. Acknowledgment

Diet-ESP is a joint work between Orange and Ludwig-Maximilians-Universitaet Munich.

11. References

11.1. Normative References

[I-D.raza-6lowpan-ipsec]
Raza, S., Duquennoy, S., and G. Selander, "Compression of IPsec AH and ESP Headers for Constrained Environments", draft-raza-6lowpan-ipsec-01 (work in progress), September 2013.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

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

[RFC2403] Madson, C. and R. Glenn, "The Use of HMAC-MD5-96 within ESP and AH", RFC 2403, November 1998.

[RFC2404] Madson, C. and R. Glenn, "The Use of HMAC-SHA-1-96 within ESP and AH", RFC 2404, November 1998.

[RFC3566] Frankel, S. and H. Herbert, "The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec", RFC 3566, September 2003.

[RFC3602] Frankel, S., Glenn, R., and S. Kelly, "The AES-CBC Cipher Algorithm and Its Use with IPsec", RFC 3602, September 2003.

[RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", RFC 3686, January 2004.

[RFC3948] Huttunen, A., Swander, B., Volpe, V., DiBurro, L., and M. Stenberg, "UDP Encapsulation of IPsec ESP Packets", RFC 3948, January 2005.

[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.

[RFC4494] Song, JH., Poovendran, R., and J. Lee, "The AES-CMAC-96 Algorithm and Its Use with IPsec", RFC 4494, June 2006.

[RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, June 2006.

[RFC4835] Manral, V., "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 4835, April 2007.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

11.2. Informational References

[RFC5856] Ertekin, E., Jasani, R., Christou, C., and C. Bormann, "Integration of Robust Header Compression over IPsec Security Associations", RFC 5856, May 2010.

Appendix A. Comparison

This section compares the proposed Diet ESP with 6LoWPAN ESP [I-D.raza-6lowpan-ipsec] related to IoT use cases. It shows the different ESP packet sent with the two compression methods. In each case the maximum possible compression is used and the underlying UDP header is compressed as much as possible. The big advantage of Diet ESP compression removing the UDP header appears. Furthermore there are no additional compression configuration bytes to be sent in each packet, like done in 6LoWPAN compression, because the configuration is negotiated at the beginning of the communication the during the IKEv2 [RFC5996] negotiation. Diet ESP uses the idea of ROHC[RFC5856] compression removing the disadvantage that the whole packet has to be sent once at the beginning of the connection, because it considers that a lot of information of the Security Association can be reused to decompress the packet.

Both comparisons are using 8 bits alignment. The figures are aligned to 16 bits to improve the readability.

A.1. Transmitting 1 Byte without anti-replay

6LoWPAN offers compression of the Sequence Number to 8, 16, 24 bit has to be sent in each packet, even if it is not going to be used.

6LoWPAN does not offer to reduce the ICV as it is not removed with NULL-authentication. Diet-ESP offers reducing to fair securely 64 bits.

AES-CTR is used for encryption.

Figure 4a and 4b show this comparison. The advantage of Diet ESP for this example is 96 bits.

```
               0                             1
               0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             0|             initialization vector             |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            16|             initialization vector             |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            32|             initialization vector             |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            48|             initialization vector             |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            64|      1 byte data      |          ICV          |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            80|                      ICV                      |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            96|                      ICV                      |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           112|                      ICV                      |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           128|          ICV          |
              +--+--+--+--+--+--+--+--+
```

Figure 4a) 1 byte Data Payload with Diet-ESP. (no SPI, no SN, no PAD, no NH)

```
               0                             1
               0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             0|     Id    | SPI |  SN |           SN          |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            16|             initialization vector             |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            32|             initialization vector             |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            48|             initialization vector             |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            64|             initialization vector             |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            80|     Id    |0 |C |  P  |source port| dest port |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            96|                     length                    |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           112|      1 byte data      |       pad length      |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           128|      next header      |          ICV          |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           144|                      ICV                      |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           160|                      ICV                      |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           176|                      ICV                      |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           192|                      ICV                      |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           208|                      ICV                      |
              +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           224|          ICV          |
              +--+--+--+--+--+--+--+--+
```

Figure 4b) 1 byte data payload with 6LoWPAN ESP. (no SPI, 8 bits SN,
8 bits pad length, 8 bits 6LoWPAN NH)

A.2. Transmitting 1 Byte to multi directional connections.

Having multiple connections to one host implies the use of the SPI to identify the correct Security Association. Diet ESP allows the reduction to 8, 16 and 24 bit. 6LoWPAN ESP offers quite the same reductions, except 24 bit. In most sensor use cases 254 possible connection are more than enough, whereas the following two pictures show the advantage of Diet ESP against 6LoWPAN ESP for an 8 bit SPI. Since there is no possibility to remove the SN with 6LoWPAN it has to be at least 8 Bit.

6LoWPAN offers compression of the Sequence Number to 8, 16, 24 bit has to be sent in each packet, even if it is not going to be used.

6LoWPAN does not offer to reduce the ICV as it is not removed with NULL-authentication. Diet-ESP offers reducing to fair securely 64 bits.

AES-CTR is used for encryption.

Figure 5a and 5b show this comparison. In case of an 8 bit SPI the advantage is 96 bits. For a 24 bit SPI the advantage would be 104 bits.

```
                0                             1
                0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
              0|          SPI          | initialization vector |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             16|             initialization vector             |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             32|             initialization vector             |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             48|             initialization vector             |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             64| initialization vector |      1 byte data      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             80|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             96|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            112|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            128|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

Figure 5a) 1 byte Data Payload with Diet-ESP. (8 bits SPI, no SN, no PAD, no NH)

```
                0                             1
                0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
              0|     Id     | SPI | SN  |          SPI          |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             16|           SN           | initialization vector |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             32|             initialization vector             |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             48|             initialization vector             |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             64|             initialization vector             |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             80| initialization vector |     Id    |0 |C |  P  |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
             96|source port| dest port |         length        |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            112|         length         |      1 byte data     |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            128|       pad length       |      next header     |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            144|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            160|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            176|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            192|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            208|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
            224|                      ICV                      |
               +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

Figure 5b) 1 byte data payload with 6LoWPAN ESP. (32 bits SPI, 16 bits SN, 8 bits pad length, 8 bits 6LoWPAN NH)

Appendix B. Document Change Log

[draft-mglt-dice-diet-esp-00.txt]: First version published.

[draft-mglt-ipsecme-diet-esp-00.txt]:
NAT consideration added.
Comparison actualized to new Version of 6LoWPAN ESP.

Authors' Addresses

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

Phone: +33 1 45 29 60 52
Email: daniel.migault@orange.com

Tobias Guggemos
Orange / LMU Munich
Am Osteroesch 9
87637 Seeg, Bavaria
Germany

Email: tobias.guggemos@gmail.com

Daniel Palomares
Orange / LIP6 - UMPC
10, Rue du Moulin
92170 Vanves, Ille-de-France
France

Email: daniel.palomares@orange.com