

# Diet-ESP: A flexible and compressed format for IPsec/ESP

draft-mglt-ipsecme-diet-esp-01.txt

**D. Migault, T. Guggemos**

25/02/2014- IETF89- London



# Table of Contents

- Securing IoT communications
- IPsec / ESP
- Diet-ESP
- Use Cases
- Diet-ESP vs. ESP
- Conclusion

# Securing IoT communications

IPsec is not widely used to secure Applications:

- No API to setup IPsec (Kernel specific)
- Applications don't want to trust the OS
- Security overhead as computed on each IP packet

(D)TLS is widely used to secure Applications:

- Common used APIs
- Application just embed the library
- Security overhead is computed on application payload

# Securing IoT communications

The IoT context:

- Sending data is very expensive (10-100 times higher than computation)
- Highly specialized device (OS + application)
- Developer specifies (or knows) the system
- Small Data usually fits in on IP frame
- IPsec can be used like a tiny Firewall, providing a secure environment for all applications

Overhead for X byte data with UDP (Security-Header + 8 bytes UDP header)

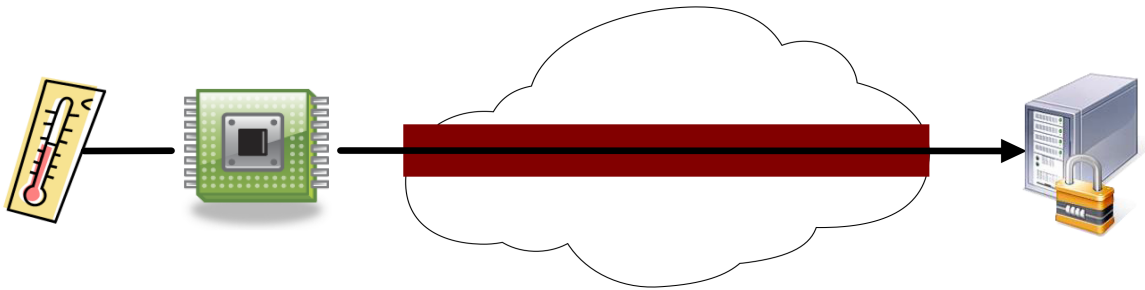
	IPsec/ESP	DTLS	Diet-ESP	6LoWPAN DTLS	6LoWPAN DTLS+UDP
<b>Packet Overhead</b>	10 + 8 bytes	13 + 8 bytes	<b>0 + 0 bytes</b>	8 + 8 bytes	8 + 3 bytes







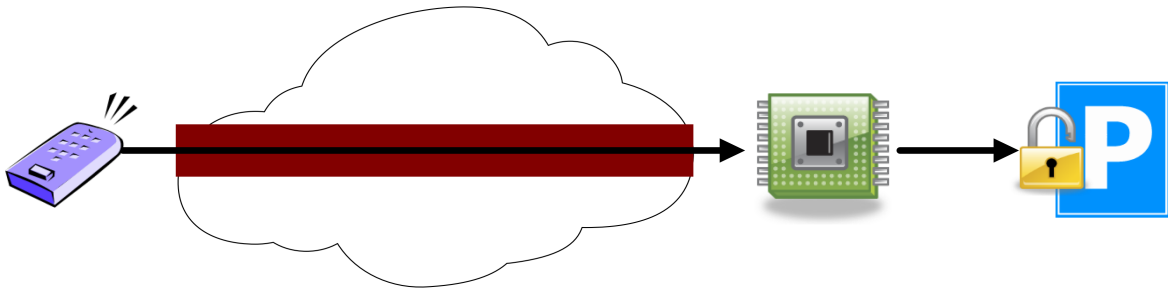
# Sending X byte of data every Y time.



- 1 application (fixed ports, specified in Traffic Selector)
  - ▶ no Next Header / no UDP-Header
- 8 bit alignment or aligned application data
  - ▶ no Padding / no Pad Length
- 1 IP connection
  - ▶ no SPI
- 1 packet per minute, no replay risk
  - ▶ no SN



# Receiving X byte of data



- 1 application (fixed ports, specified in Traffic Selector)
  - ▶ no Next Header / no UDP-Header
- 8 bit alignment or aligned application data
  - ▶ no Padding / no Pad Length
- multiple Senders with different IPs
  - ▶ no SPI
- some packets per day, replay risk
  - ▶ 1-2 byte SN

# Diet-ESP vs. ESP

## Impact on ESP

- ESP-header (compressed / removed fields)
- Encrypted ESP-payload (removed fields)
- ESP-ICV (compressed field)

## Impact on IPsec (eventually):

- SAD Lookup (variable SPI, no SPI)

## How can we make this an ESP extension?

- Diet-ESP Context is negotiated during IKE\_AUTH exchange
- Modifications applies to the negotiated SA.

# Negotiation with IKEv2

Initiator	Responder
-----	-----
1) HDR, SAi1, KEi, Ni	-->
	<-- HDR, SAR1, KEr, Nr
2) HDR, SK { IDi, [CERT,] AUTH, SAi2, TSi, TSr, N(DIET_ESP_SUPPORTED [CTX_1, CTX_2, CTX_3, ANY])}	-->
	<-- HDR, SK { IDr, [CERT,] AUTH, SAR2, TSi, TSr, N(DIET_ESP_SUPPORTED [CTX_2])}

# Security Considerations

## Reducing/Removing the SPI

- Regular SAD-Lookup
  - ▶ Limiting number of possible connection on the gateway.
- Modified the SAD-Lookup
  - ▶ higher effort to identify SA
  - ▶ DoS-vulnerability

## Reducing/Removing the SN

- Prevents/Downgrade anti-replay-protection
- Can we use ESN mechanism without sending the all lower 32 bit on the wire?

## Reducing the ICV

- Reducing level of Security, provided by the MAC algorithm.

# Accomplished Work / Next Steps

## Accomplished Work:

- Development Demonstration in Python (ca. 400 lines of code)
- Rudimentarily Contiki implementation
  - ▶ based on the ESP implementation of Vilhelm Jutvik
  - ▶ less than 100 lines of code patches
- Start performance measurements

## Next Step:

- Deployment on Excelyo
- Alternate Algorithm (Encryption / Authentication)
- IV reduction

Thank you for your attention

Looking for a PhD  
tobias.guggemos@gmail.com

# SAD Lookup

The SPI size is unknown for incoming packets.

→The lookup for incoming ESP packets, has to be changed:

Naive implementation:

one lookup for each supported SPI\_SIZE (in general 5)

- perform 5 lookups based on IP\_D, IP\_S, SPI
- if more than one SA fits:
  - ▶ authenticate with all found keys
  - ▶ use the SA with the fitting key

→ DoS vulnerability

# SAD Lookup

## Solution:

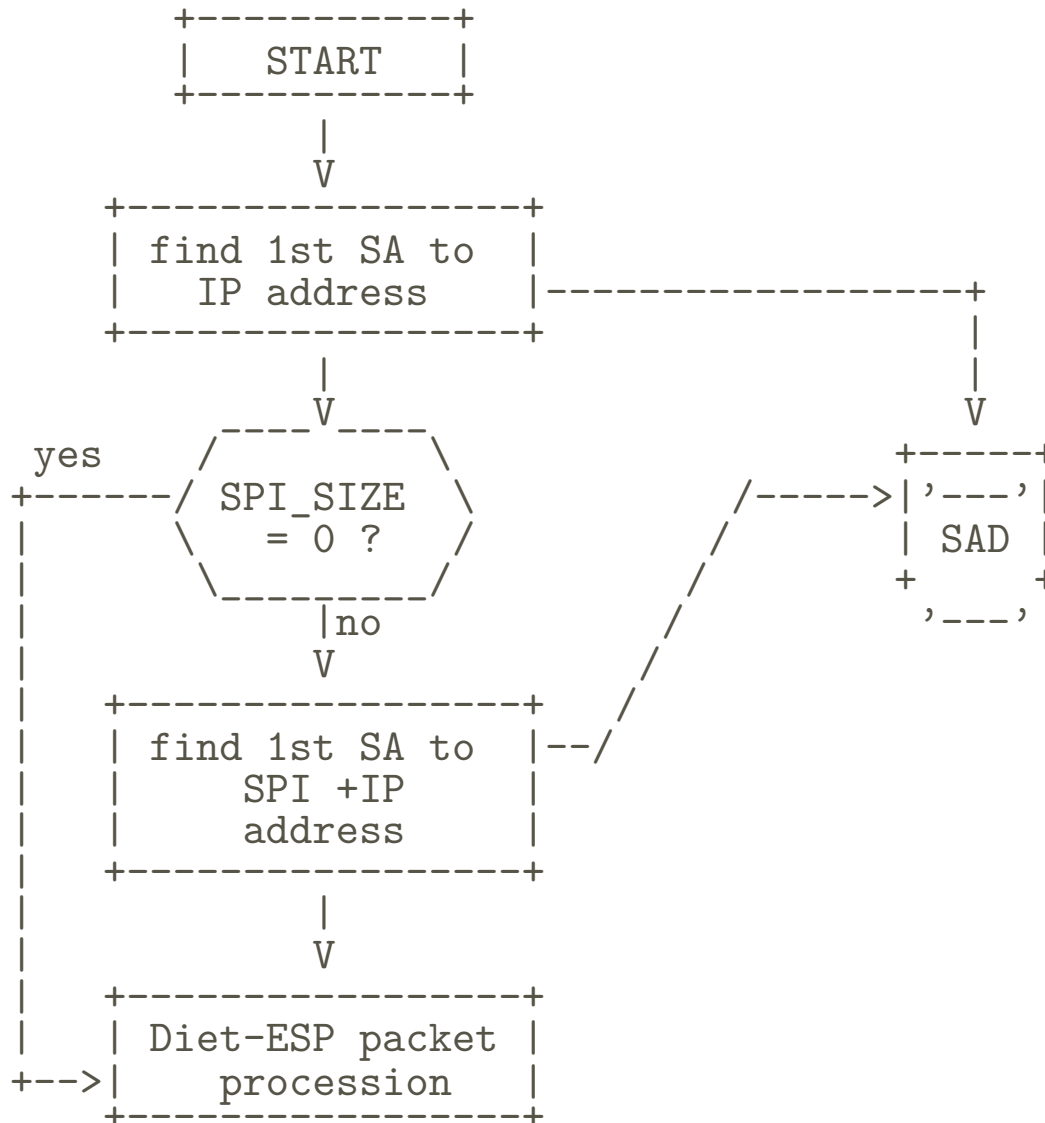
- Forbid overlapping SPIs for one IP\_D – IP\_S – combination
- Impossible for Mobile IP

2 possibilities:

- ▶ code the SPI size in the SPI field (1-4 byte)
  - has to be included to the SA negotiation
  - impossible for 0 SPI
  - reduces SPI range
- ▶ IP address based lookup



# IP address based lookup



# Compression

Possible compression:

- ESP-header: 8 byte → 0 byte
- ESP-trailer: 2 (+ PADDING) byte → 0 byte
- ESP-payload:
  - ▶ Transport Mode UDP: 8 byte → 0 byte
  - ▶ Tunnel Mode IPv4: 20 byte → 0 byte
  - ▶ Tunnel Mode IPv6: 40 byte → 0 byte
  - ▶ Tunnel Mode 6LoWPAN: 2-42 byte → 1 byte

# Compression

How is that possible?

- Most informations are already stored in the Traffic Selectors:
  - ▶ IPsec mode
  - ▶ IP addresses
  - ▶ Transport protocol
  - ▶ Transport protocol ports (cannot be a range)
- only the mapping is difficult:
  - ▶ reduced interoperability
    - but in general, sensors have exactly ONE function
  - ▶ Application Developer knows the intended use
    - he can specify the possible compression