Diet-ESP: A Flexible and Wide Range Security protocol

draft-mglt-6lo-diet-esp-requirements draft-mglt-6lo-diet-esp

D. Migault, T. Guggemos, S. Raza, C. Bormann

16/07/2016- IETF96- Berlin

Motivations

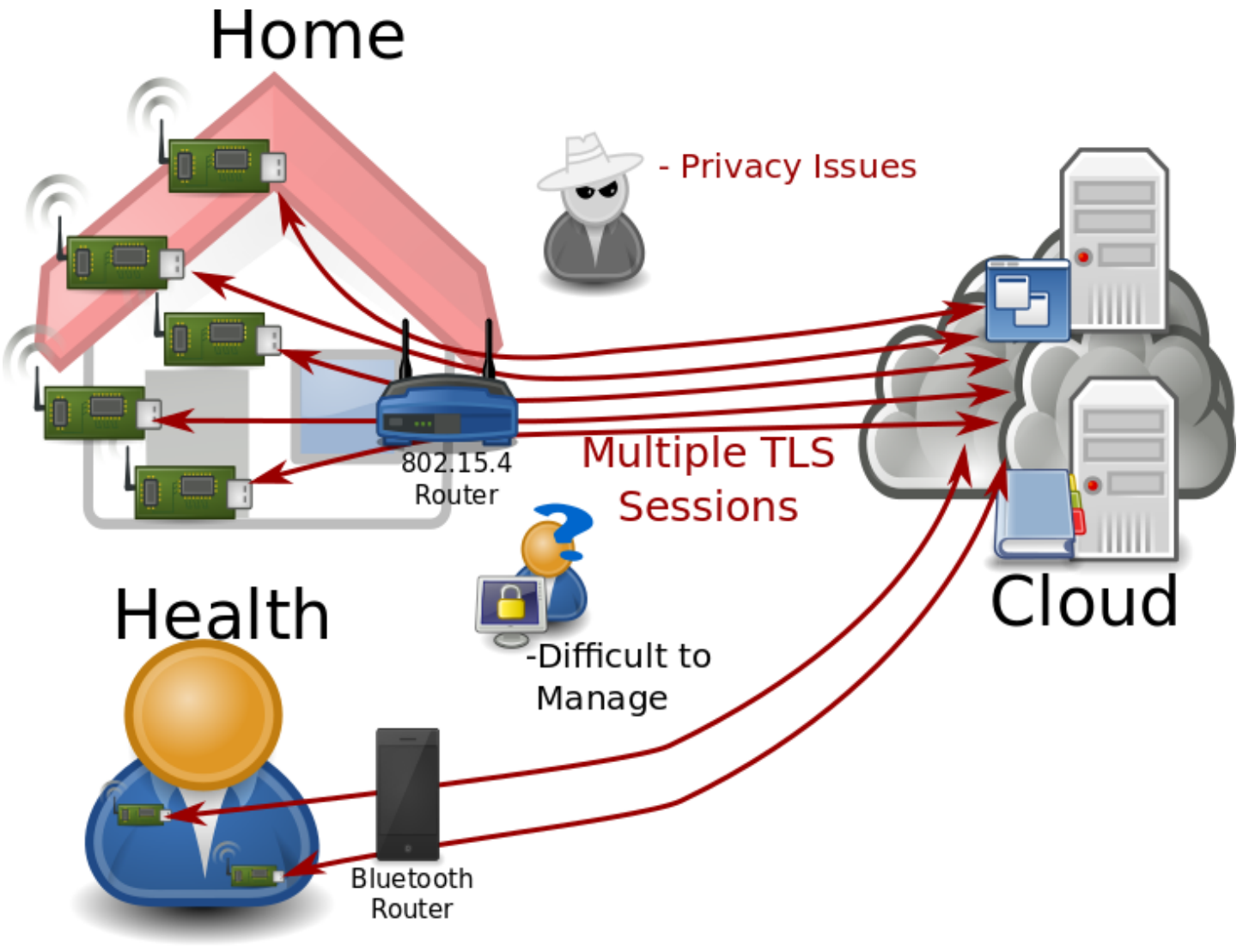
The current de-facto IoT security protocol is DTLS1.2 (DICE profile).

- Reasonable choice for:
  - Web based IoT applications,
  - End-to-end security
- but in our view DTLS1.2/DICE does not address all IoT segments

The presentation shows:

- E2E security does not address all cases
- IPsec/ESP addresses new scenarios
- Diet-ESP an optimized IPsec/ESP for IoT

E2E Security with (D)TLS



E2E Security: Problem Statement

Privacy:

- Leaks information on the destination service:
  - IP, port, data frequency, number of probes
  - for Health service, this could typically identify the disease
- For probes configured once for ever:
  - No control on the delivered data by the owner
  - No control the destination service

Management Issues for (probes configured once for ever):

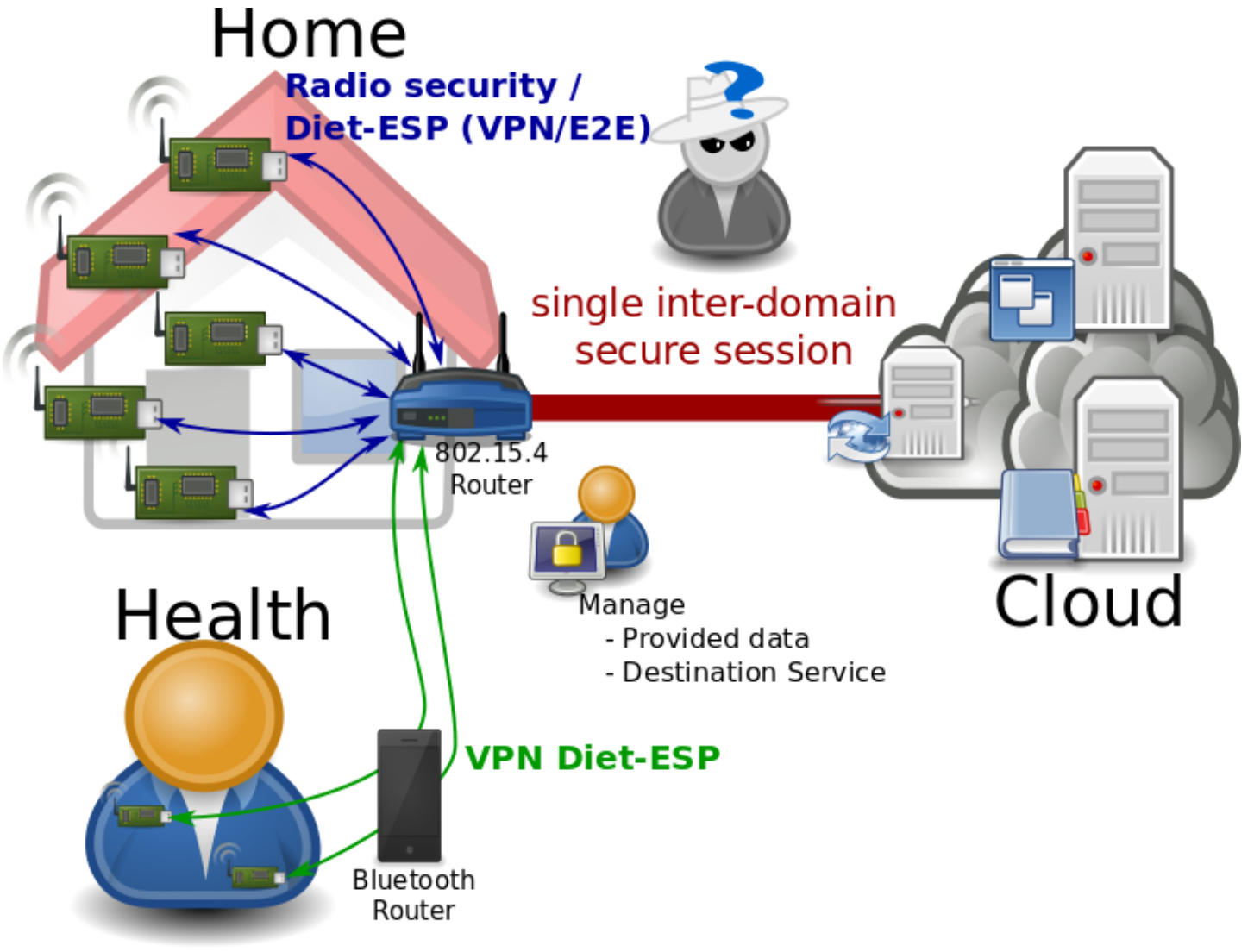
- Data policies, destination service is performed by:
  - Replacing probes
  - Re-configuring probes one-by one

E2E Security: Problem Statement

Architecture:

- Many key management operations:
  - Each sensor set a TLS Session with a unique session key
- Bottleneck architecture
  - No possibility to define intermediary boundaries between probe and application service
    - Prevent the propagation of an attack
    - Distribute the network / compute load
    - Traffic aggregation
  - Corrupted / misconfigured probes unnecessarily load the data center
  - Centralization of many idle TLS Sessions
  - No possibility of multicasting

Secure Domain with ESP



Scenario Conclusion

There are significant scenarios that shows:

- IPsec/ESP complements (D)TLS/DICE

ESP got significant features that justify to have ESP for IoT

ESP Features

Security & Privacy :

- Secures the transport layer
- Avoids disclosure of the application (port)/ sensor involved (IP)

Flexibility:

- ESP is independent of key exchange protocols
  - IKEv2
  - DEX
  - GDOI [RFC6407] (Multicast)
  - Other (3GPP)

ESP Features

Scalability:

- Extends device-to-server to device-to-domain communications
- Enables Multicast communication [RFC4046,RFC6407]
- Re-use existing know-how in secure infrastructures deployments:
  - Mobile networks manage millions of devices

Good integration with 802.15 radio layer:

- Re-uses IKEv2 / DEX key exchange protocols implemented in radio chips

Interoperability:

- A SINGLE version over more than 20 years
- A SINGLE version for all transport protocols

Diet-ESP in 1 Slide

ESP already has a low overhead compared to DTLS but Diet-ESP does even better

Diet-ESP is based existing standards:

- ESP is an envelop carrying encrypted data (ESP Header / ESP Trailer)
- ROHC

Diet-ESP defines:

- ROHC-like Profiles for a flexible and reduced security overhead
- ROHC-like light compressor

Diet-ESP Position:

- Encryption: NO compression of ANY cryptographic content
- 6LoWPAN / Diet-ESP:
  - Diet-ESP compresses ESP Header AND/OR ESP Trailer

Network Overhead Comparison

PROTOCOL_OVERHEAD (AES-CCM with ICV of 8 bytes)

- Network Overhead (Bytes)

DTLS/DICE	ESP	Diet-ESP
29	20	8 - 20

- Radio Frame (Bytes)

802.15.4	LORA	BlueTooth	SIGFOX
102	> 59	> 23	12

Conclusion

Current Status:

- draft-mglt-6lo-diet-esp-requirements provides ESP requirements for IoT
- draft-mglt-6lo-diet-esp describes Diet-ESP that addresses these requirements
- Diet-ESP has been implemented and tested on Contiki
  - Adapting ESP to Diet-ESP took a few days

Why 6lo WG ?

- Works with any L2 that supports 6lo stack
- Provides Network-level security for all data
  - Between Gateways
  - Between Host nodes
  - For all applications and control data
  - Required for high security IoT deployment

Next Steps

* Is the WG interested to work on Diet-ESP?

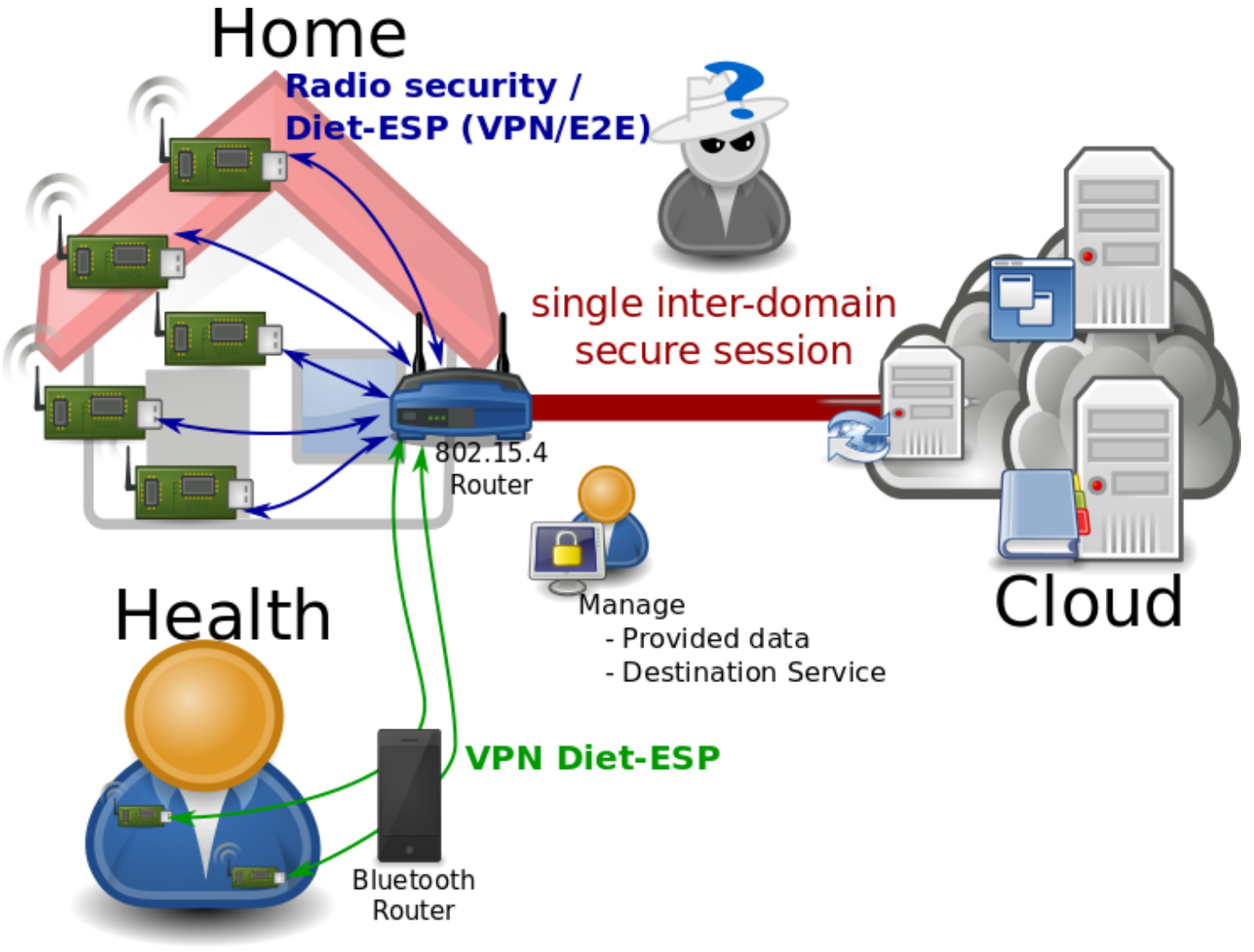
Thank you for your attention

Annex

This provides additional material:

- A1 ESP Scenarios:
  - Scenario 1: Secure Domain
  - Scenario 2: Key exchange independent
  - Scenario 3: Secure Multicast
- A.2 ESP vs TLS
  - DICE Packet Description
  - ESP Packet Description
  - Architecture Comparison
- A.3 ESP Multicast
- A.4 Diet-ESP

Scenario 1 Secure Domain with ESP



Scenario 1 Secure Domain with ESP

In addition to E2E IoT should be provided a security protocol that defines domains:

- To scale large deployment of probes
- To better protect privacy
- To provide the data owner the ability to manage its data / service

IPsec/ESP provides:

- E2E security
- The definition of a security domain

Scenario 1 Secure Domain with ESP

Single inter-domain secure session:

- Protects data, transport, port
- Limits privacy leakage to the Cloud destination
- Reduces the number of sessions between the Cloud and the probes
- Ease the IoT model of shadowing probes in the cloud

Home Domain:

- Enables to delegates load, aggregation
- Enables data owner to manage its sensors / application / cloud provider
- Takes advantage of the radio layer security

Scenario 2: Key exchange protocols independent

3GPP TR 33.863 describes the IoT use case, where:

- Radio Resource are allocated with keys between the two peers
- An additional key exchange (IKEv2/TLS) is not necessary

Small devices with limited capacities may use adapted key exchange protocol:

- Ex. DEX is use on devices that do not have the resource required for IKEv2

Problem Statement:

- TLS/DICE comes with a mandatory key exchange

IPsec/ ESP does not defines any protocol exchange and can be used with:

- IKEv2
- DEX
- Manually configured

Scenario 3: Multicast

Multicast enables to send a given piece of information to multiple probes. Potential scenarios include:

- Software updates
- Configuration of the sensor
- Providing a command to multiple probes

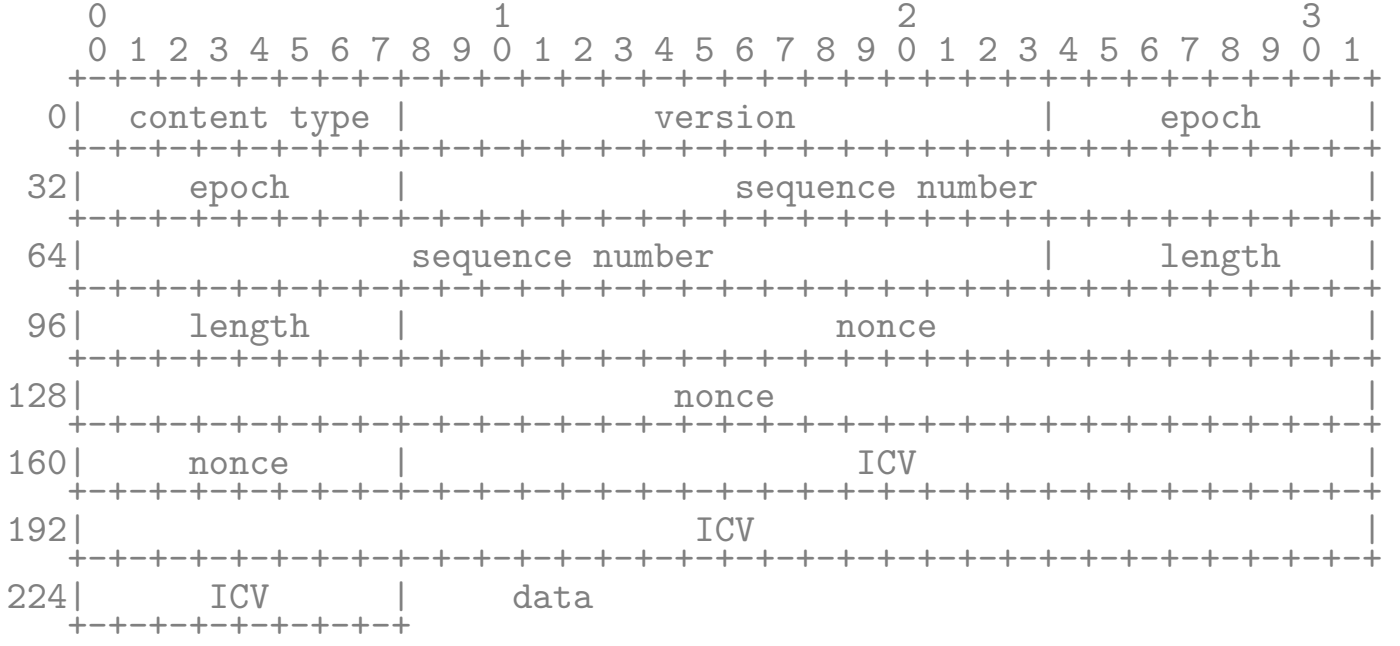
The advantage of using multicast is that:

- It saves bandwidth a one message is shared by n nodes
- It speeds the propagation to all nodes making one-to-many communications real time
- Authentication / encryption uses symmetric cryptography
  - Authentication of the session is done once for the life time of the session

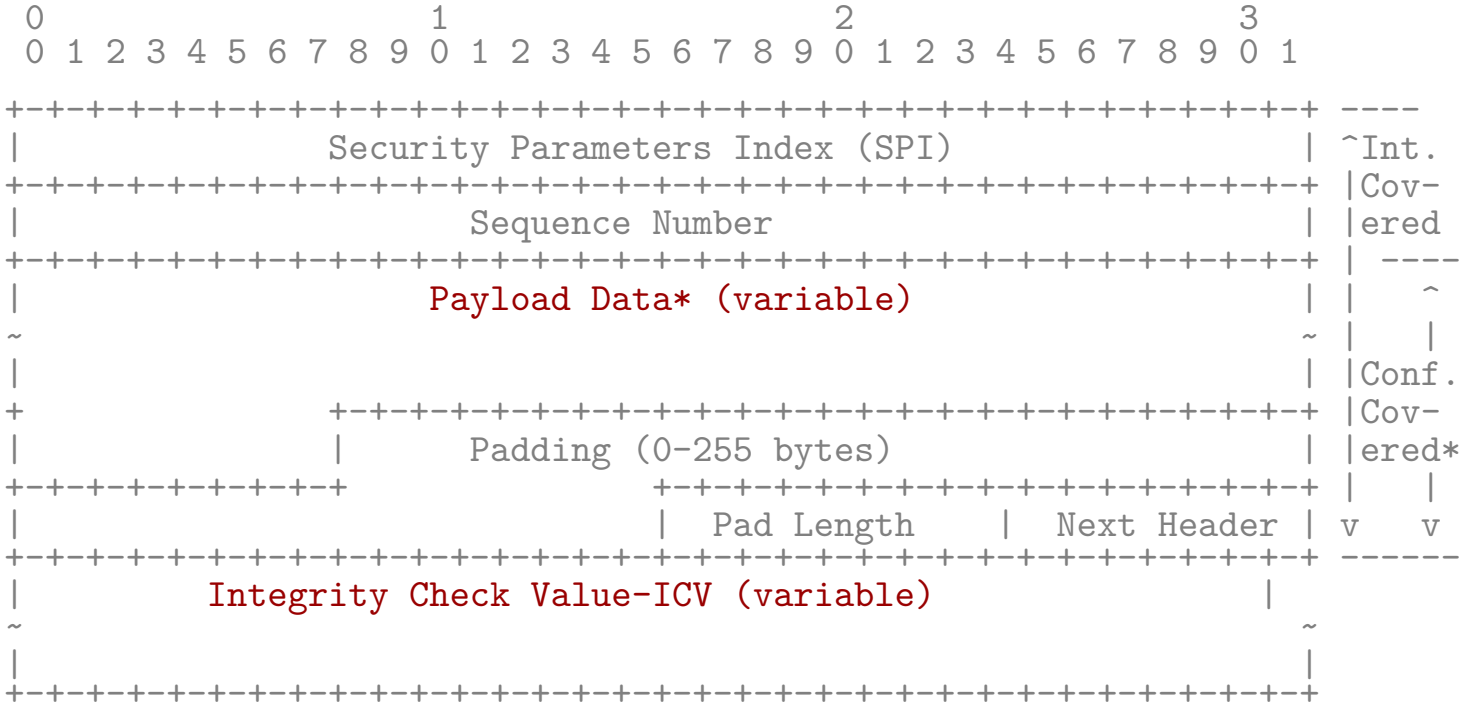
Problem Statement:

- (D)TLS/DICE does not provide multicast extensions, ESP does [RFC4046,RFC6407].

DICE packet Description



ESP packet Description



Architecture differences between ESP/TLS

The main architecture differences between ESP and TLS/DTLS are:

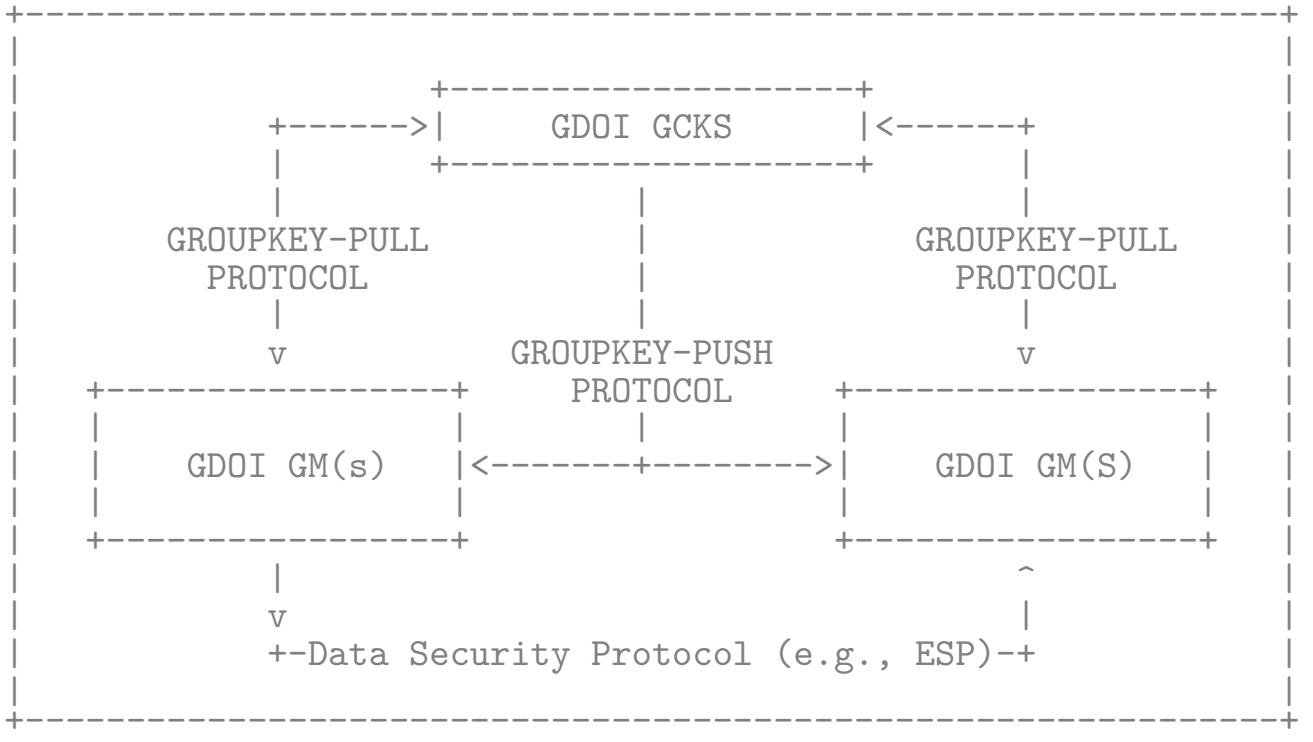
- ESP protects IP packets (including UDP/TCP)
- TLS/DTLS protects the TCP/UDP Payload
- ESP is implemented in the kernel
  - makes application kernel-specific (interfacing with kernel APIs)
- TLS/DTLS is a library
  - Makes application development OS-independent

Providing OS independent application explains the success of TLS/DTLS:

- In IoT the distinction between kernel / user space is not so constraining

Group Key Management

- Interesting for securing multicast
- RFC6407:



Diet-ESP

Diet-ESP aims at compressing:

- ESP Header
- ESP Trailer

Diet-ESP is not something new:

- ROHC/6LowPAN provides ways to compress the ESP Header
- ROHCoverIPsec provides ways to compress the clear text data

What is new with Diet-ESP is that:

- It enables to compress Padding, Pad length and Next Header fields
  - Compression occurs before the encryption (similarly to ROHCoverIPsec)
  - Compression occurs after the ESP fields are added (unlike ROHCoverIPsec)
  - There is no decompression of these fields. (one way compression)
- It takes advantage of IKEv2 to agree on the compression rules
  - Results in a light-compression framework

Diet-ESP

Currently Diet-ESP is based on ROHC:

- ROHC Compressor/decompressor
  - With initialized states
  - Without synchronization, initialization exchanges
- ROHC profiles

Open Issues are:

- Does Diet-ESP defines ROHC compressor or a specific compressor ?
  - Other alternatives exists like SCHC
  - Our profiles will be called rules
- Can we consider the profiles as an extension of ROHCoverIPsec ?
  - This would make possible ROHC/Diet-ESP and Diet-ESP

Diet-ESP

Diet-ESP is based on ESP:

- Diet-ESP is based on ESP
- Diet-ESP is able to send ESP packet
  - Enable natural fall back to uncompressed ESP
  - Preserve ESP interoperability
- Diet-ESP DOES NOT modify cryptographic parameters, algorithms
  - Crypto is left untouched