DTLS-based Multicast Security for Low-Power and Lossy Networks (LLNs)

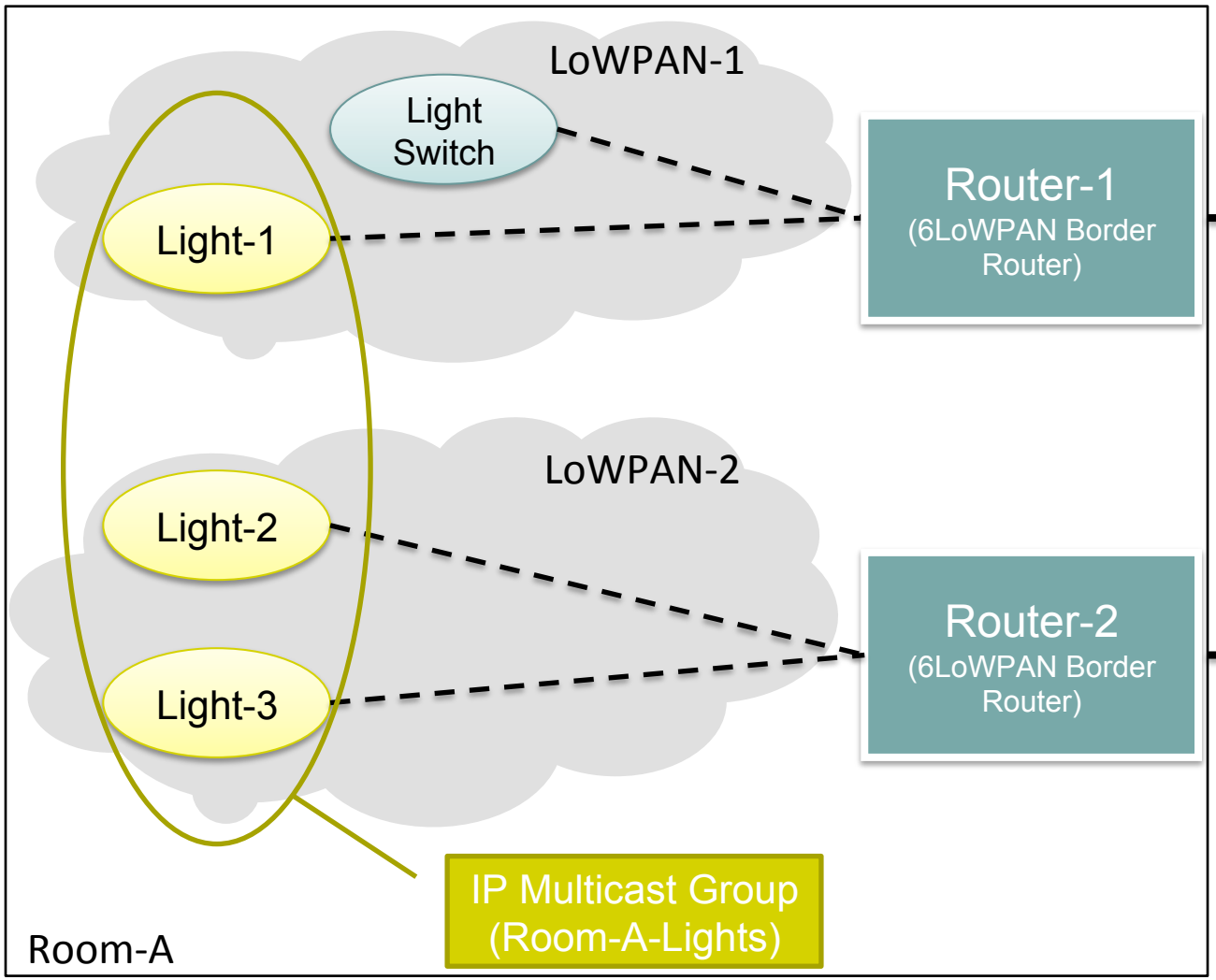
draft-keoh-dice-multicast-security

Sandeep S. Kumar, *Sye Loong Keoh, Oscar Garcia-Morchon, Esko Dijk*

IETF88 Nov 4, 2013, Berlin

Email: sandeep.kumar AT philips.com

Group Communication Use Cases



Source: Group Communication for CoAP (draft-ietf-core-groupcomm)

Use-cases

- Group light control
- Firmware updates
- Parameter distribution

Motivation & Requirements

Group communication (in LLNs): also vulnerable to eavesdropping, tampering, message forgery, replay, etc.

Limited resources and memory: reduce the number of cryptographic protocols on device.

DTLS is chosen security solution for unicast CoAP: beneficial for constrained devices if it can be used also for COAP group communication.

Requirements

Goals of this draft

–Group level data integrity and authentication

–Data confidentiality (optional)

–Replay protection

Out-of-scope (possible future draft)

–Data source authentication: *application level*, e.g., object security

–A Group Security Association (GSA): *distribute keying materials*, *specify the ciphersuite for encryption and authentication*

–Multicast key management: *update/renew group keys periodically.*

Reuse of DTLS Record Layer

Assumptions:

•Group Security Association (group session key and cipher for authentication & encryption to use) are known to all group members **out-of-band**.

•Multiple senders and multiple listeners/receivers in the group communication.

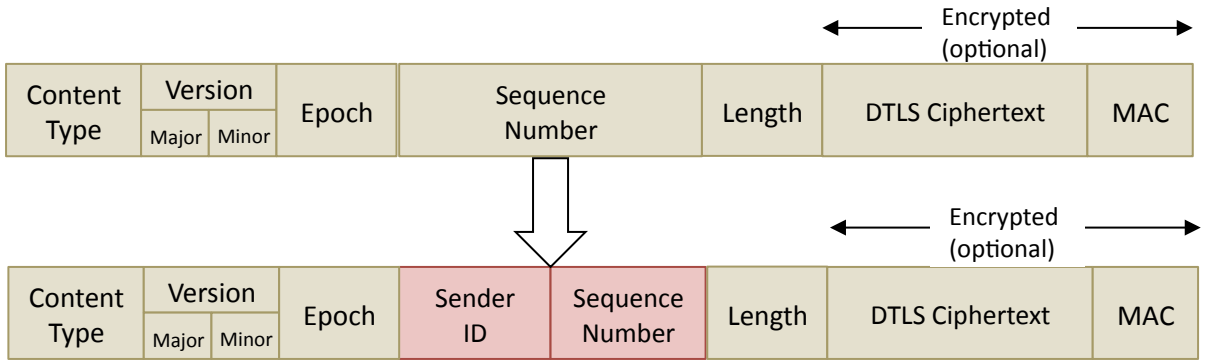
Proposal:

•Each sender gets a *unique* ***SenderID (2-byte)*** either chosen by a controller or randomly or derived from the IPv6 address

– Fallback mechanism if *Sender-ID's* are not unique

•In the DTLS Record Layer, split the ***6-byte*** *sequence number* field into:

2 bytes *Sender ID* and ***4 bytes*** *"truncated" sequence number*.

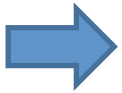


draft-keoh-dice-multicast-security

Why split sequence number?

- Reuse of nonce breaks security of CCM and GCM modes of operation (AEAD ciphersuites in TLS)
- In (D)TLS

```
struct {
    opaque salt[4];
    opaque nonce_explicit[8];
} CCMNonce;
```



```
struct {
    uint32 server_write_IV; // low order 32-bits
    uint64 seq_num; // TLS sequence number
} CCMServerNonce.
```

- In DTLS specifically

  64-bit sequence number = 16-bit epoch||48-bit seq_num

- If multiple senders send messages with the same key
  - Either synchronize seq. number => hard in practice
  - Provide separate seq. number space for each sender => our approach

Protecting Group Messages (1)

GSA is set out-of-band

=> DTLS SecurityParameters are set for all senders and listeners

All devices derive (or provided out-of-band) the same six DTLS key material

client write MAC key

server write MAC key

client write encryption key

server write encryption key

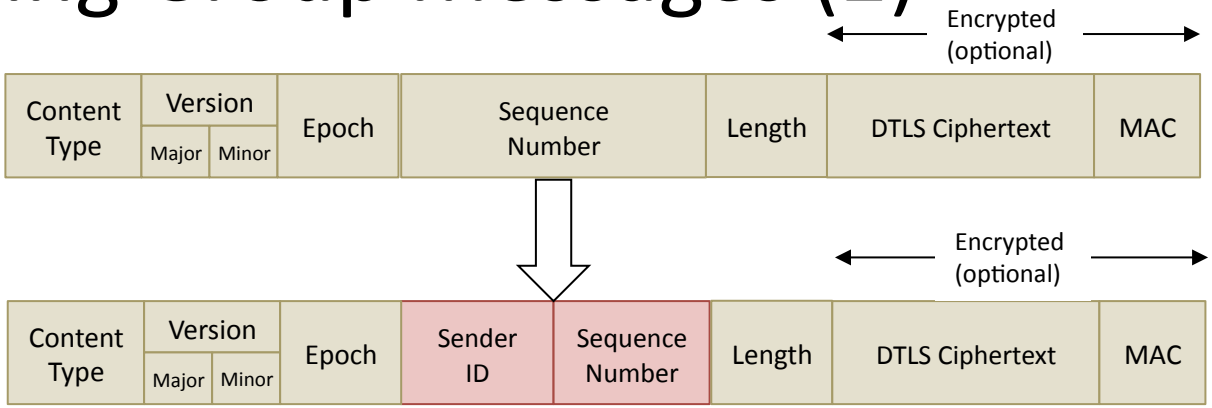
client write IV

server write IV

For Senders : *SecurityParameters.ConnectionEnd="server"*

For Listeners : *SecurityParameters.ConnectionEnd="client"*

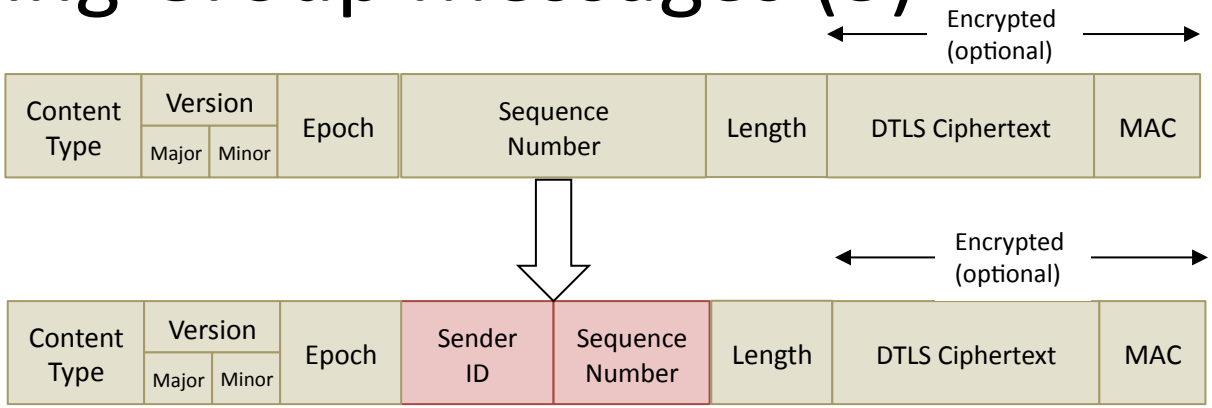
Protecting Group Messages (2)



Senders

- "write state" is instantiated with "server write" parameters.
- Each sender manages its own *epoch* and "truncated" *sequence number*
  - no synchronization is needed with other senders in the group. Initialized to 0.
- The sender include its *Sender ID* in the DTLS Record Layer header and increments the "truncated" sequence number when sending a group message.
- The *epoch* will be increased, and the "trunc." *sequence number* will be reset once the group session key is renewed or updated (**out-of-scope: to be defined as part of key management**)

Protecting Group Messages (3)



Listeners

- Multiple "read states" are instantiated with "server write" parameters for each sender linked by *SenderID*
  - Keying material same but the epoch and the "truncated" sequence number of the last received packets needs to be kept different for different senders.
- Listeners use the *multicast destination IP address* of the packet to lookup the "server write" key.
- Message is decrypted and the MAC of the message is checked
- Using the *Sender ID* field, receivers retrieve the last used *epoch* and *sequence number* to detect replayed messages.
  - If success: update last seen seq number from the SenderID in the "read state"

Other issues (1)

- Epoch update and change cipher spec security
  - Sequence number wraps need to be handled as part of key management (out of scope)
- Late joiners (John Foley)
  - Use technique similar to AERO (Authenticated Encryption with Replay prOtection)
  - First seen packet used to initialize the epoch&seq number but **drop it**. Check replay of next messages.
  - What if a chain of packets are being replayed?

Other issues (2)

- *SenderID*s are not unique
  - Fallback: All senders are also listeners to the group
  - If sender sees a message from a different device with the same *SenderID* then stop using SenderID
  - Contact controller and inform about clash -> controller provides new *SenderID*s to one or both
- Use specific ciphersuite suitable for multicast
  - **AERO** (Authenticated Encryption with Replay prOtection) (*draft-mcgrew-aero*) mode provides inbuilt mechanism to support multi-senders
  - Should it be mandated as the only ciphersuite for DTLS multicast or keep it flexible to support all existing AEAD modes?

Summary

- Group communication is often used in machine-to-machine (M2M) applications.
- Group communication is equally vulnerable and requires security.
- Preferably re-use existing security protocols on constrained devices in LLNs.
- Propose to reuse DTLS Record layer to support secure group communication, with key management **out-of-scope**.

Next Steps

- Is this draft ready to be adopted as a WG document?