EAP-based Authentication Service for CoAP

draft-ietf-ace-wg-coap-eap-03

Rafael Marín-López, University of Murcia
Dan García-Carrillo, University of Oviedo

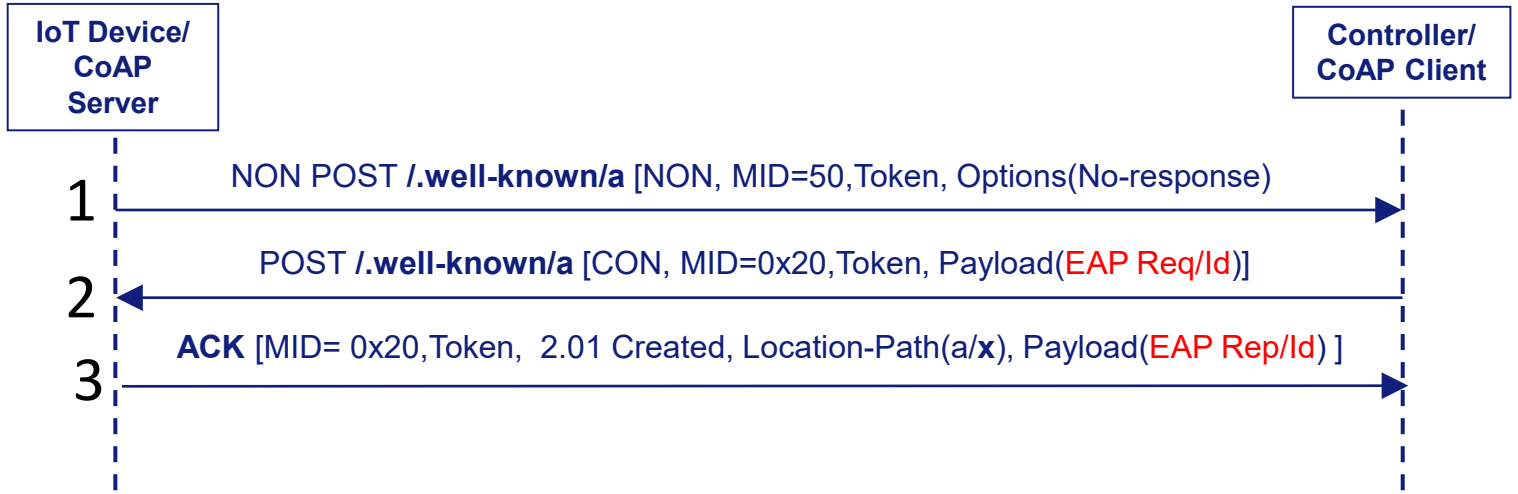
IETF 111 meeting, July 29th, 2021

CoAP-EAP – Updates Summary of 03 version

- Added /.well-known/ for both entities
- Changed URI to conform to HATEOAS
- Added Error handling section
- Added Cryptosuite negotiation
- Elaborated process of key derivation
- IANA considerations

CoAP-EAP – Added /.well-known/ for both entities

- Added well-known URI **/.well-known/a (To be assigned by IANA)**
- Set in both entities, IoT device and Controller



CoAP-EAP – Ordering guarantee following HATEOAS

- In the first ACK the server can choose the value of the URI as it pleases.

Example

- The server can create a new resource with structure
  - /a/x
    - x -> Value representing the current step in the authentication process
  - Could be a completely different value */randomValue*

- The CoAP engine will take care of handling retransmissions, duplicate detection, sending error for non-existing resources, etc.

CoAP-EAP – Error handling

Possible Issues - How to manage out of place POST /.well-known/a ?

From EAP authenticator to peer

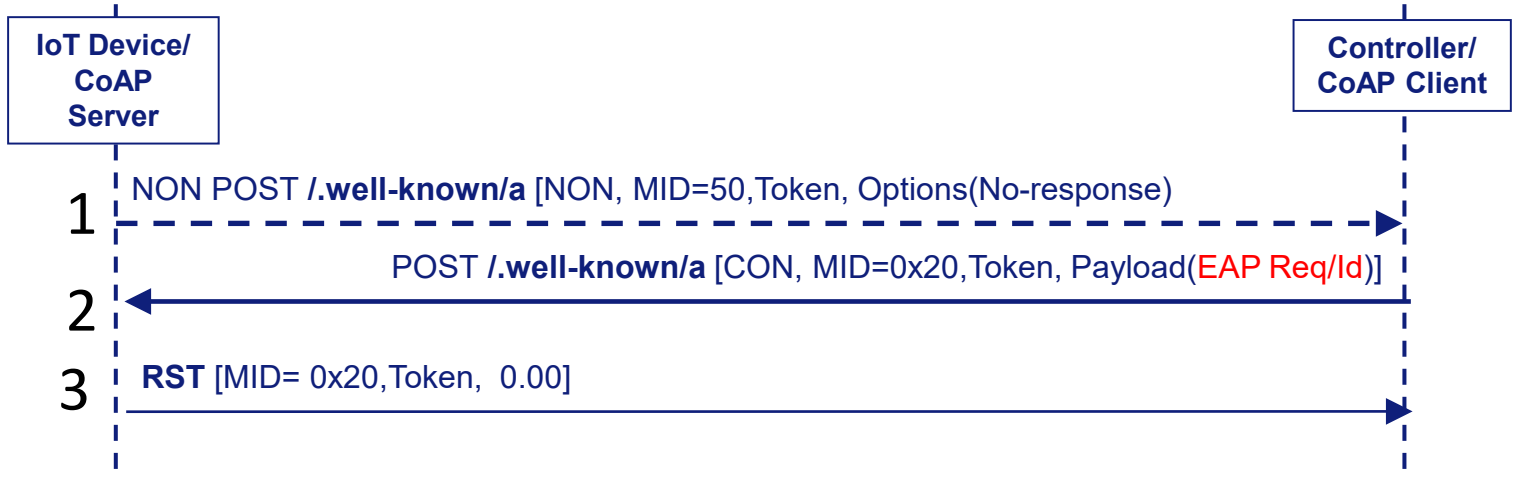
WITH or WITHOUT ONGOING Authentication

- Send a CoAP Reset message. The IoT device did not send the starting message

CoAP-EAP – Error handling

From EAP peer to authenticator

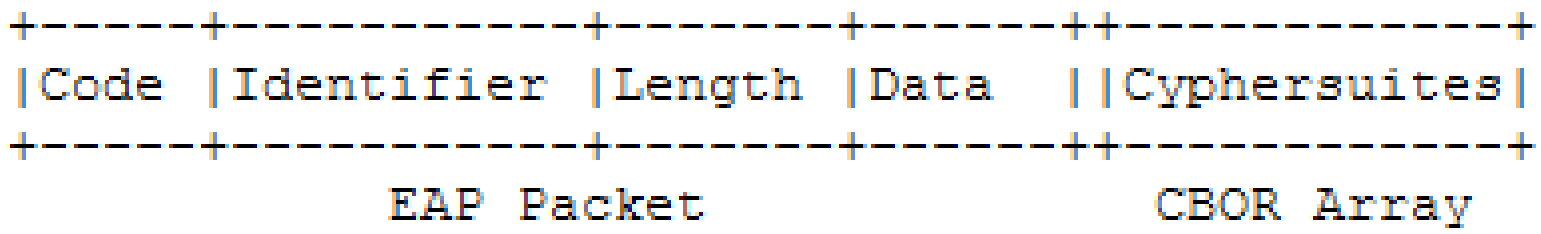
- WITH ONGOING Authentication
  - OMIT since the message is NON Confirmable with No-Response Option
- WITH no ONGOING Authentication
  - If arrives to the CoAP-EAP application in the Controller, tries to start.
  - Being out of place, the IoT did not purposely send this message, sends Reset.



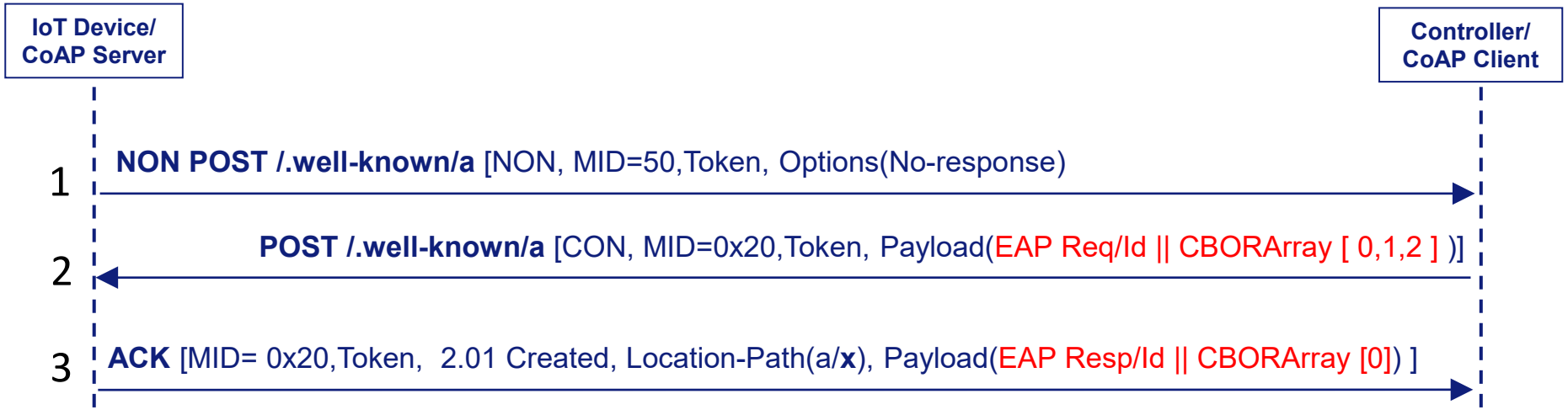
CoAP-EAP – Cyphersuite negotiation

- How to manage the Cyphersuite negotiation within the existing exchange (Not adding more messages)
  - New Option
    - Not our first choice because
      - All CoAP implementations should be updated
      - It will only be used for CoAP-EAP, it is not something useful in other CoAP application
  - Embedding the cyphersuite negotiation in the CoAP payload
    - A cleaner option as we do not modify existing implementations
    - Only need a defined structure to parse.
- The cyphersuite negotiation is embedded into the key derivation to bind them and prevent a downgrading attack.

CoAP-EAP – Cryptosuite negotiation



Example of disposition of the CoAP Payload



Exchange with the cryptosuite negotiation

CoAP-EAP – Key derivation

Cyphersuites compatible with OSCORE

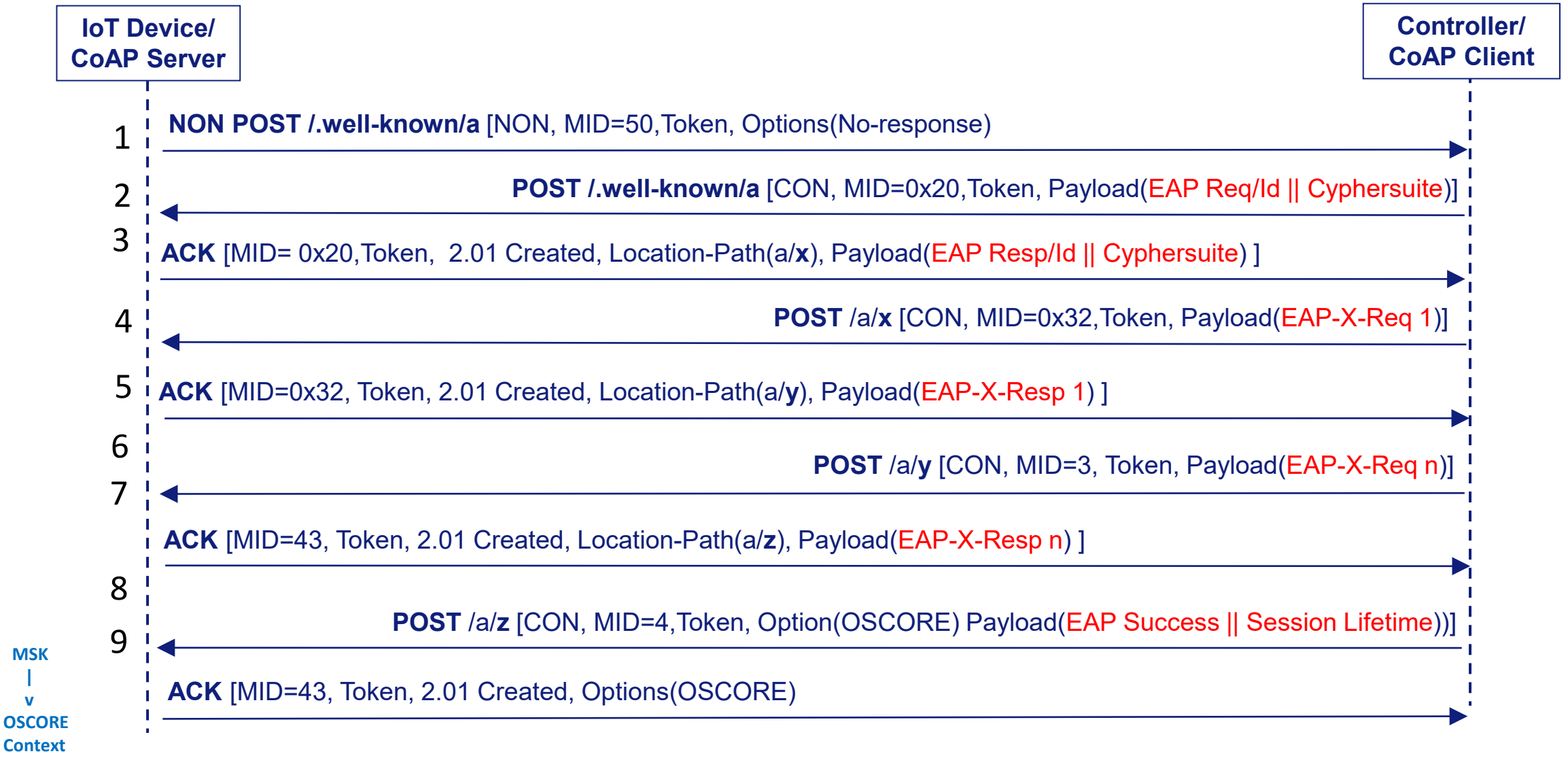
	AEAD	**, HASH**
0.	AES-CCM-16-64-128	, SHA-256
1.	A128GCM	, SHA-256
2.	A256GCM	, SHA-384

- Master Secret = KDF(MSK, CSO | "OSCORE MASTER SECRET" , length)
- Master Salt = KDF(MSK, CSO | "OSCORE MASTER SALT" , length)
- Recipient ID = KDF(MSK, "OSCORE RECIPIENT ID" , length)
- Sender ID = KDF(MSK, "OSCORE SENDER ID" , length)

Where:

- KDF is the HKDF-Expand function from (HMAC)-based key derivation function (HKDF) defined in [RFC5869]
- MSK is the Master Session Key derived from the EAP method
- CSO is the concatenated content of the Cyphersuite negotiation. If empty the null-string is used.
- labels are specific for each derivation
- Length is the max length of the output key material. Each one as a specific maximum length specified by OSCORE.

CoAP-EAP – Current state



IANA considerations

- Assignment of EAP lower layer identifier
- Assignment of the URI /.well-known/a

THANK YOU