

# Post-Quantum and Hybrid enhancements for EAP-AKA'

<https://datatracker.ietf.org/doc/draft-ietf-emu-hybrid-pqc-eapaka/>  
<https://datatracker.ietf.org/doc/draft-ietf-emu-pqc-eapaka/>

IETF 125 Shenzhen

**Aritra Banerjee (Nokia)**

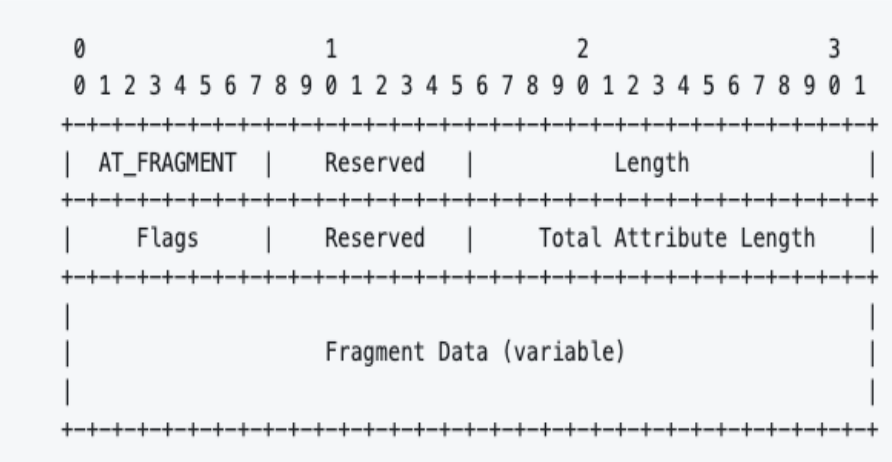
K Tirumaleswar Reddy (Nokia)

# Since IETF 124 Montreal

- The latest version inserts a new section on Message Fragmentation and Reassembly
- It also adds a new section on Capability Negotiation
- Hybrid PQC draft refers to these new changes made in the pure PQC draft to avoid duplicity.

# Message Fragmentation

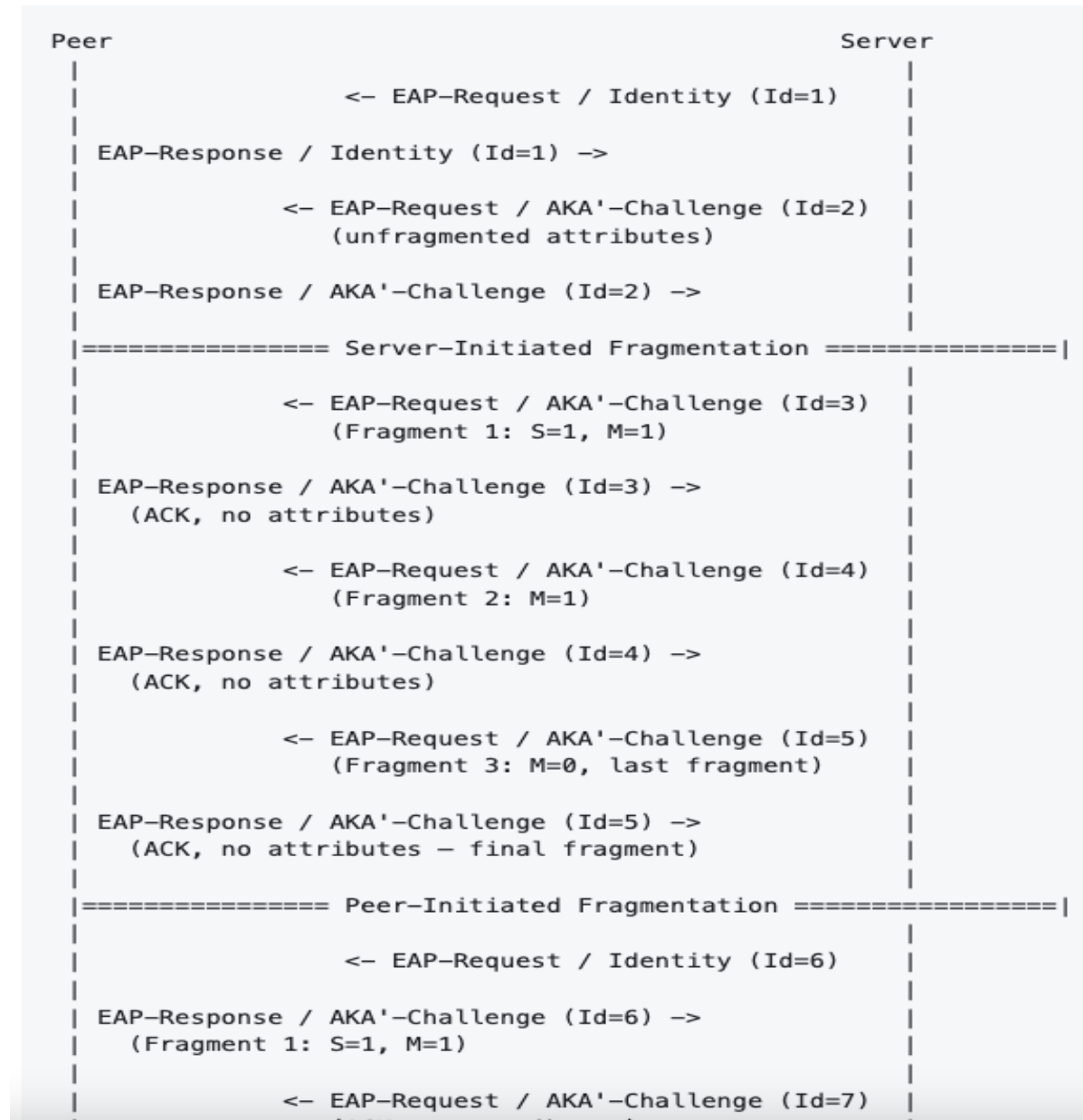
- Explicitly discusses that EAP-AKA / EAP-AKA' FS do not define native attribute-level fragmentation and reassembly, and that PQ public keys and ciphertexts may exceed EAP MTU, so the spec defines explicit attribute-level fragmentation.
- Defines **AT\_FRAGMENT** and a detailed format including:
  - “Flags” with **S** (first fragment) and **M** (more fragments) bits
  - “Total Attribute Length”
  - “Fragment Data”: carries the fragmented attribute
  - Reassembly requirements (bitwise identical reconstruction).



# Message Fragmentation and EAP ID

- Adopts a lock-step acknowledgment model like EAP-TLS, where:
  - When the receiver gets a fragment with  $M=1$ , it responds with an EAP packet of the same type with no attributes as an acknowledgment.
  - The sender must not send the next fragment until the ack is received.
- The EAP Identifier field is used to correlate fragments and acknowledgments:
  - The Identifier in an EAP-Response **MUST** match the Identifier of the immediately preceding EAP-Request.
  - Fragment acknowledgments **MUST** echo the Identifier of the fragment being acknowledged.
  - Retransmitted fragments **MUST** reuse the same Identifier value as the original transmission.
  - For fragmented exchanges initiated by the EAP server, the Identifier in each EAP-Request carrying a fragment **MUST** be incremented relative to the previous EAP-Request.

# Fragmentation Sequence Diagram



# Reassembly

- The receiver **MUST** reassemble attribute fragments strictly in the order received and **MUST NOT** process the fragmented attribute until all fragments have been successfully received and validated.
- The final fragment is identified by the M bit being cleared ( $M = 0$ ). The receiver **MUST** acknowledge each fragment, including the final fragment, using the lock-step procedure defined for fragmentation. The sender **MUST** wait for acknowledgment of the final fragment before considering the fragmented attribute exchange complete.
- During the fragmentation phase (i.e., while the M bit is set in AT\_FRAGMENT), the EAP peer **MUST** respond to each EAP-Request/AKA'-Challenge fragment with an EAP-Response/AKA'-Challenge message containing a zero-length attribute payload. These responses serve solely as transport-level acknowledgments and **MUST NOT** trigger any AKA' cryptographic processing.

# Reassembly and Security

- The EAP peer **MUST NOT** initiate USIM processing (e.g., passing RAND and AUTN to the USIM) while attribute fragmentation is in progress. USIM processing has to occur only after the final fragment (M = 0) has been received and the complete attribute set has been successfully reassembled and validated. At that point, the peer will invoke the AKA' algorithm using the RAND and AUTN values contained in the reassembled message.
- If a fragment is lost or corrupted, normal EAP retransmission procedures apply. Retransmitted fragments **MUST** use the same EAP Identifier value as the original transmission.
- The receiver **MUST** verify that:
  - The first fragment has the S bit set and subsequent fragments do not; and
  - The cumulative length of all received Fragment Data fields equals the Total Attribute Length.

This version also clarifies that fragmentation does not change AT\_MAC calculation rules (AT\_MAC covers what's on the wire including AT\_FRAGMENT), and reassembled processing occurs after MAC verification.

# Capability Negotiation

- Support for PQC KEM is negotiated using the AT\_KDF\_FS attribute.
- AT\_PUB\_KEM and AT\_KEM\_CT use an extended attribute header format that is incompatible with legacy EAP-AKA and EAP-AKA' implementations because of KEM and public key sizes for PQC Algorithms
- These attributes MUST NOT be sent unless a mutually supported PQC KEM has been successfully negotiated via AT\_KDF\_FS.

# Next Steps

- Comments and suggestions are welcome
- Ready for WGLC