Qualcomm

Updates Relevant to UE Centric 5G Security WP2.0

Zhimin Du Ph.D.
Director, Technical Standards
Qualcomm

Contents

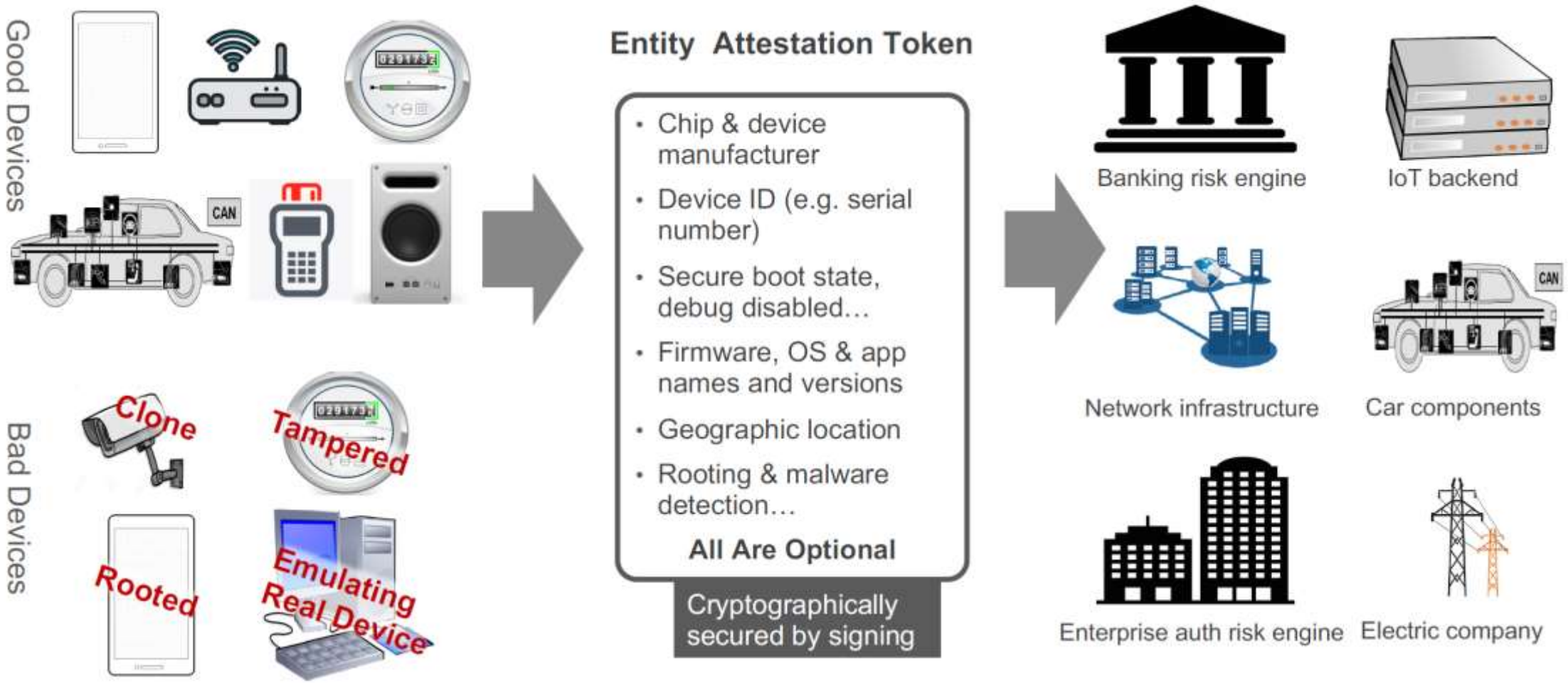
- EAT for Hardware Token based Remote Attestation
- AKMA for 5G Application Key Management

EAT: Entity Attestation Token

- Being developed in IETF RATS (Remote ATtestation ProcedureS) WG
- Core function
  - An Entity Attestation Token (EAT) provides a signed (attested) set of claims that describe state and characteristics of an entity, typically a device like a phone or an IoT device. These claims are used by a relying party to determine how much it wishes to trust the entity
- In WG Draft status, target for a new RFC
  - https://datatracker.ietf.org/doc/draft-ietf-rats-eat/
  - Last updated in this Feb.

- Open issues, public review comments and potential open source implementation being tracked in GitHub
  - https://github.com/ietf-rats-wg/eat/

Why EAT?


Good Devices
CAN
Bad Devices
Clone
Tampered
Rooted
Emulating Real Device
Entity Attestation Token
• Chip & device manufacturer
• Device ID (e.g. serial number)
• Secure boot state, debug disabled…
• Firmware, OS & app names and versions
• Geographic location
• Rooting & malware detection…
All Are Optional
Cryptographically secured by signing
Banking risk engine
IoT backend
Network infrastructure
CAN
Car components
Enterprise auth risk engine
Electric company

EAT: Core Designs

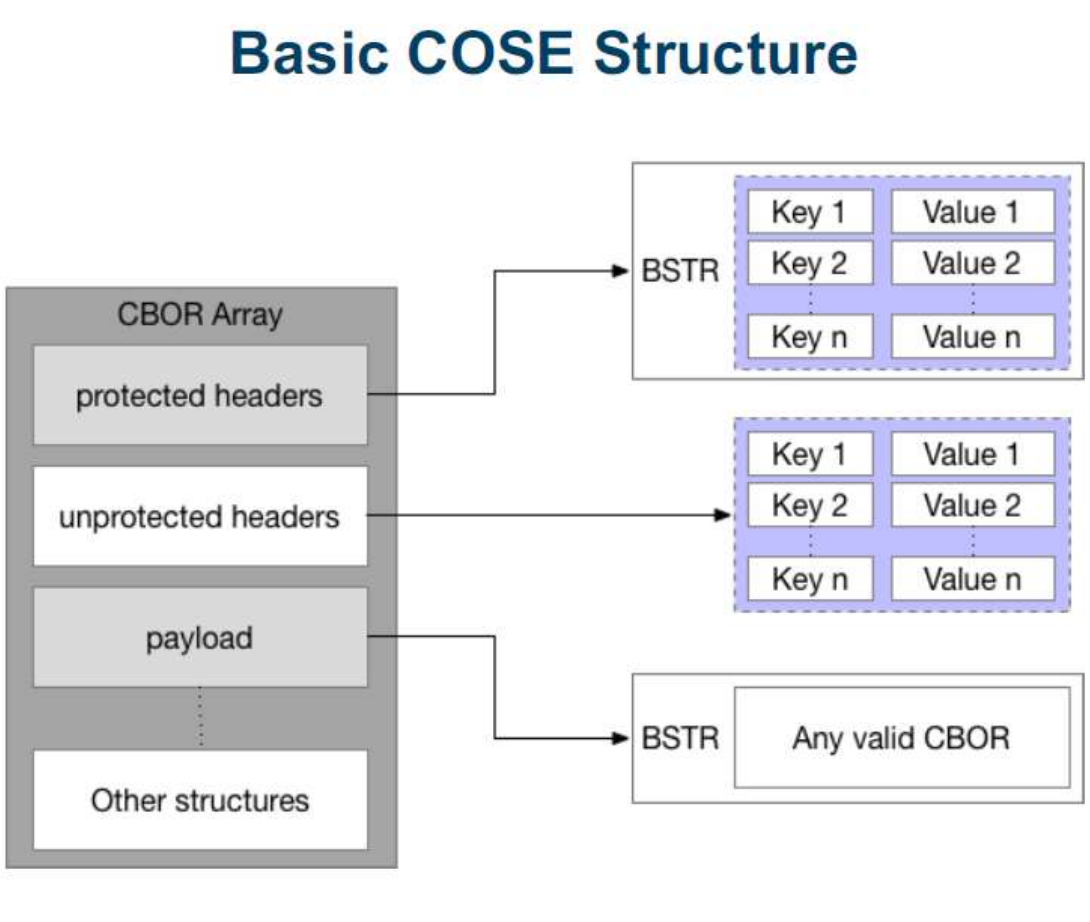
- **Overall**
  - An extensible and crypto-agile container for transporting Claims about a device state
  - Built on the IETF CDDL (Concise Data Definition Language) over IETF COSE (CBOR Object Signing and Encryption) or JSON data structures standards
  - Inherits from CWT (CoBR Web Token), later also adds JWT
    - Both administrated by IANA

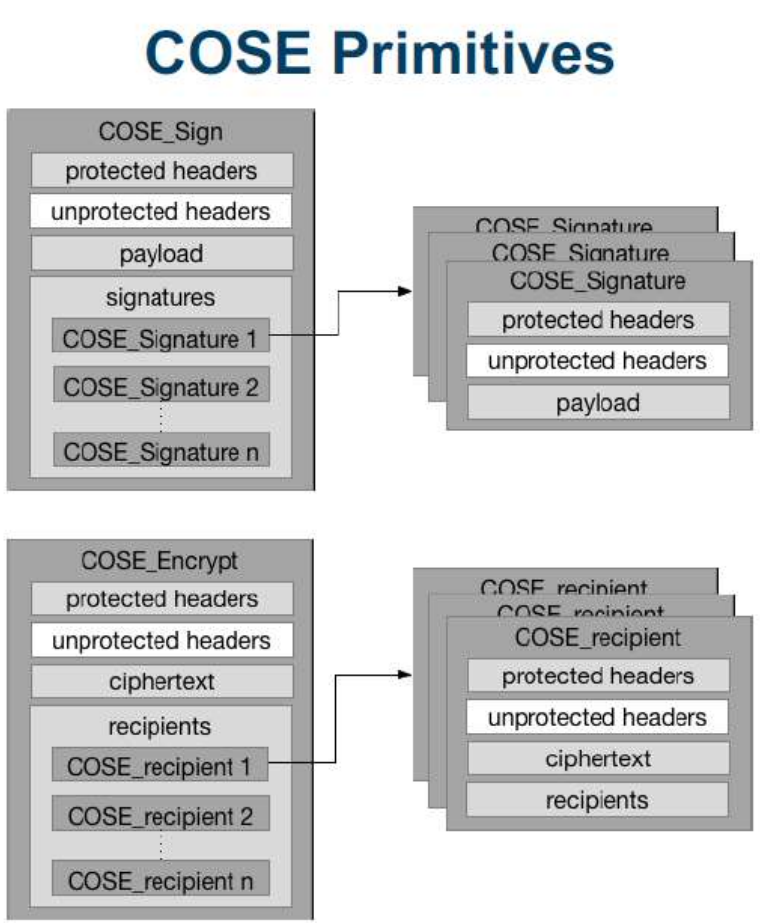
- **Claims**
  - Token ID, Timestamp, Nonce, Universal Entity ID, Origination, OEM ID, Security Level, Boot State, Location, Age, etc.
    - All optional.
  - Also extensible to add in other claims
    - Via CWT registry through IANA

EAT: Core Designs (cont.)

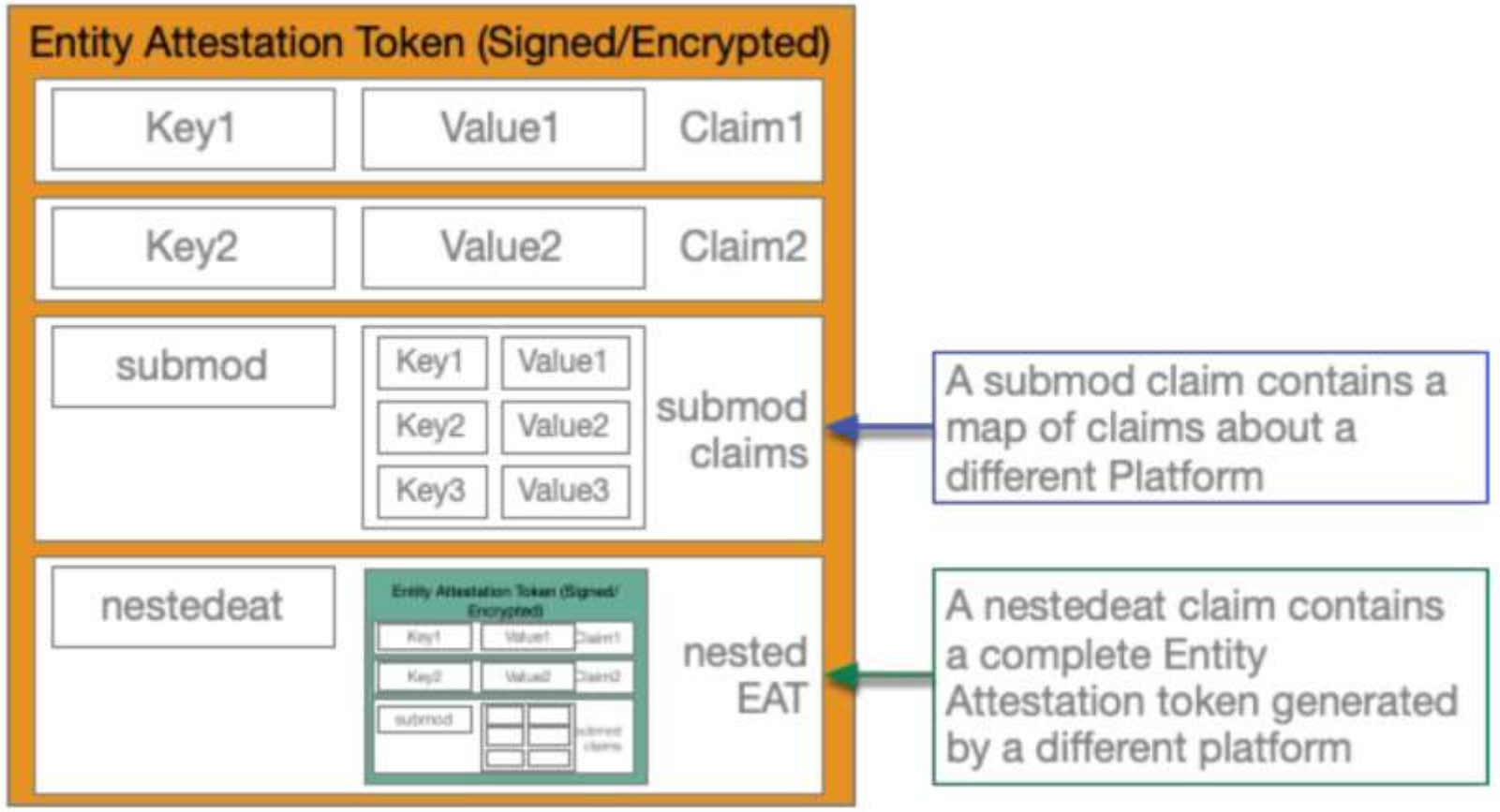
- **Flexible Usage**
  - A top-level token is associated with a device – a finished commercial end product
  - A device may have a set of submodules
    - Examples: WiFi subsystem, DSP subsystem
    - A submodule has a set of claims of its own
    - One level of submodules – keep it simple
    - The security of a submodule is either the same or less than that of the device
  - Tokens may be nested
    - This allows submodules that have attestation keys to create their own attestations

EAT Foundation: COSE



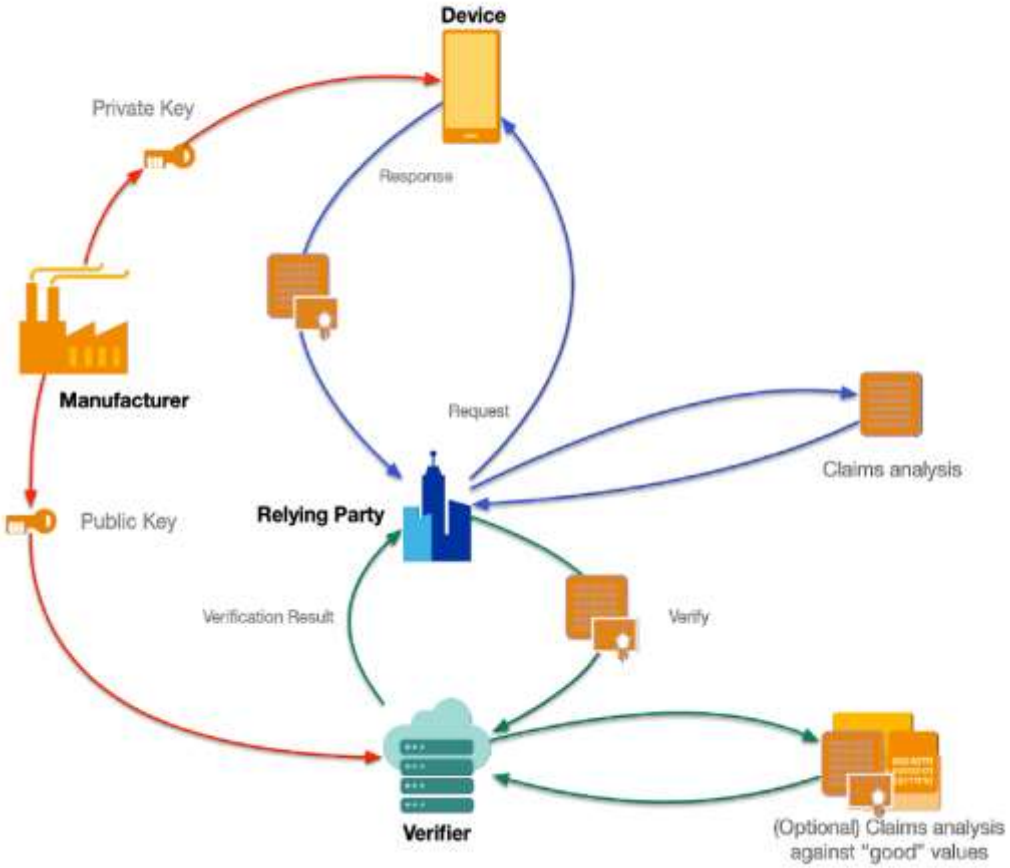


EAT Example

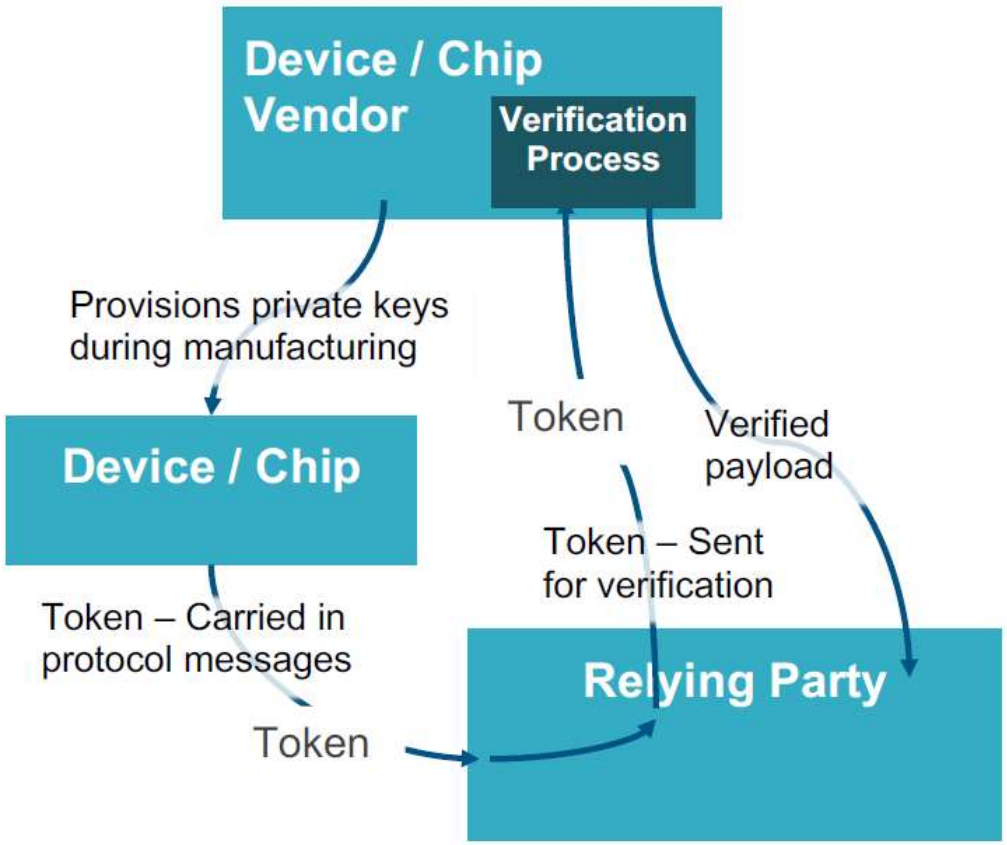

Entity Attestation Token (Signed/Encrypted)
Key1
Value1
Claim1
Key2
Value2
Claim2
submod
Key1
Value1
Key2
Value2
Key3
Value3
submod claims
nestedeat
nested EAT
A submod claim contains a map of claims about a different Platform
A nestedeat claim contains a complete Entity Attestation token generated by a different platform

How EAT works

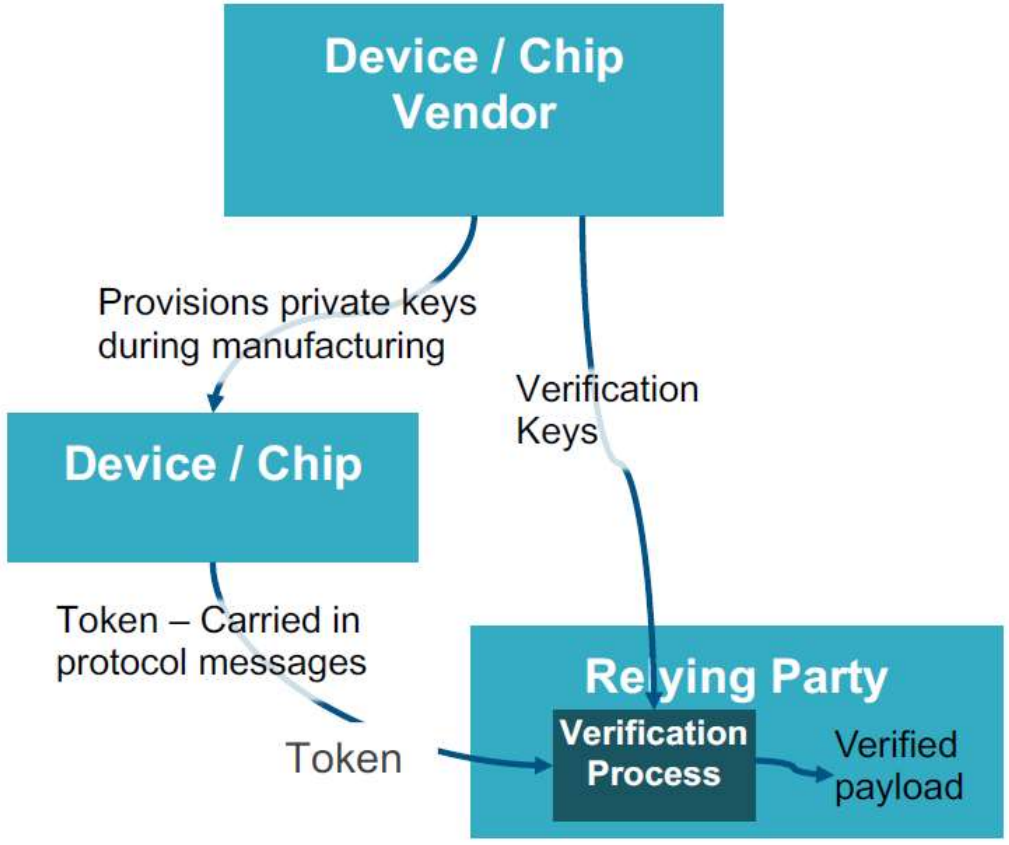
- A combination of Hardware Token and Remote Attestation
  - Keys securely created at manufacture
    - Ensure a device can be identified
  - Relying party requests an attestation
    - e.g. Nonce, Authorization, Requested Claims
  - Device signs (optionally encrypts) attestation
    - Includes claims about its state, freshness, etc.
  - Relying party uses a verifier to check the attestation
    - Is the device what is claims to be?
    - Is the device properly configured?
  - Verifier returns information about the attestation
    - Many forms of verifier – depends on needs, scale, ecosystem



How EAT works (cont.)



1. Device / Chip Vendor Provides a Service



2. Device / Chip Vendor Provides Keys

Contents

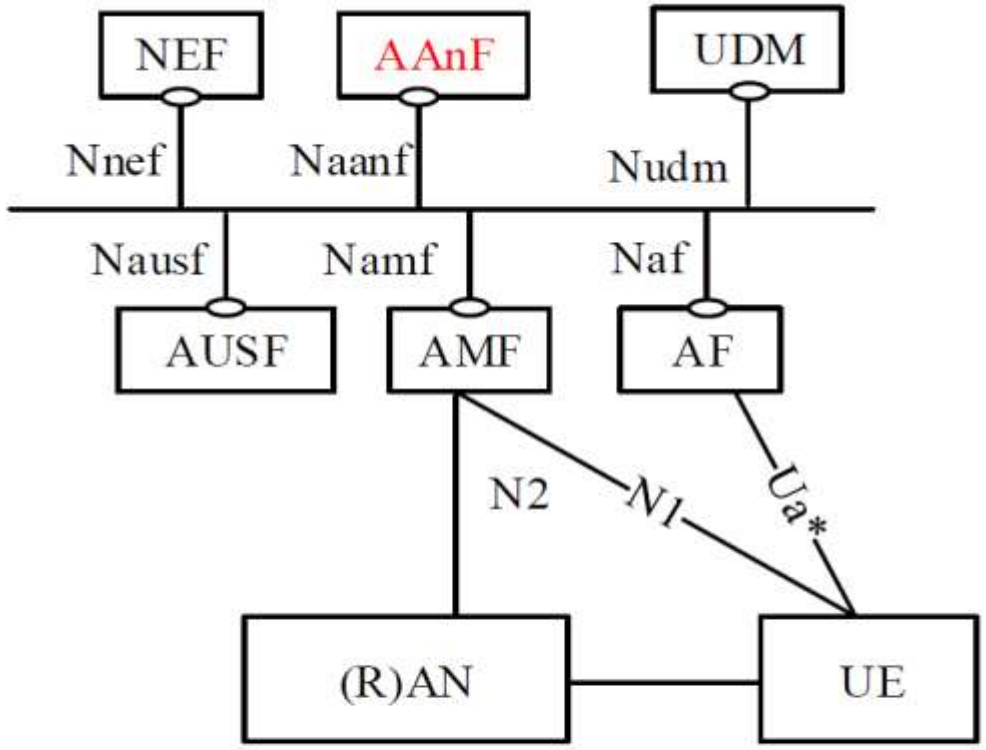
- EAT for Hardware Token based Remote Attestation
- AKMA for 5G Application Key Management

AKMA: Authentication and Key Management for Applications

- An R16 WI just completed by 3GPP SA3
  - Specified in TS33.535
  - Leverages the existing 3GPP subscription credentials & the keys resulting from the 5G primary authentication to bootstrap shared secrets (keys) to secure access to applications
  - Can be considered as “an optimized version of GBA for 5G”

- SA2 and CT relevant spec changes to be done in R17

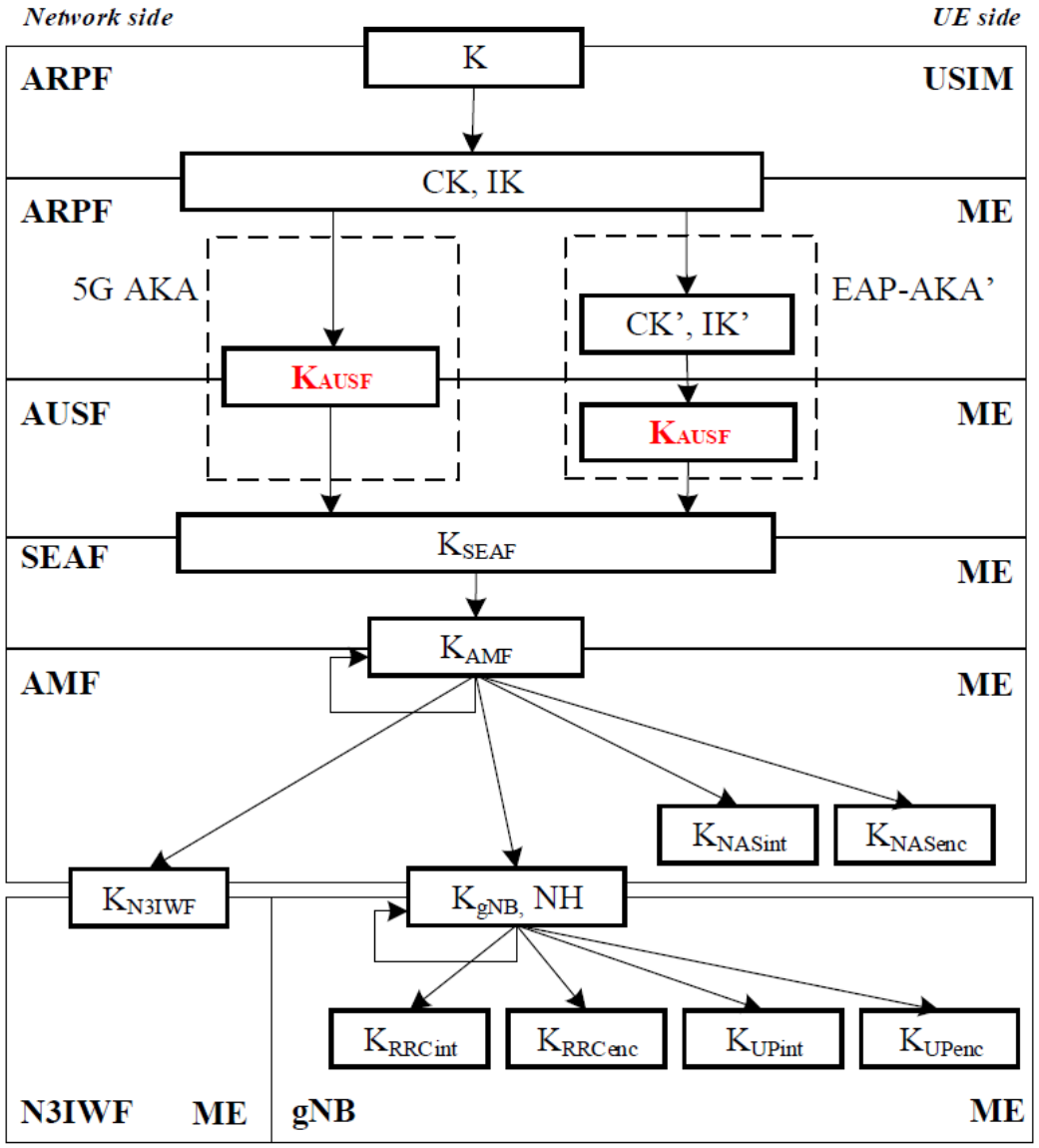
AKMA: System Architecture



UDM : Unified Data Management
AUSF: Authentication Server Function
NEF: Network Exposure Function
AF: Application Function

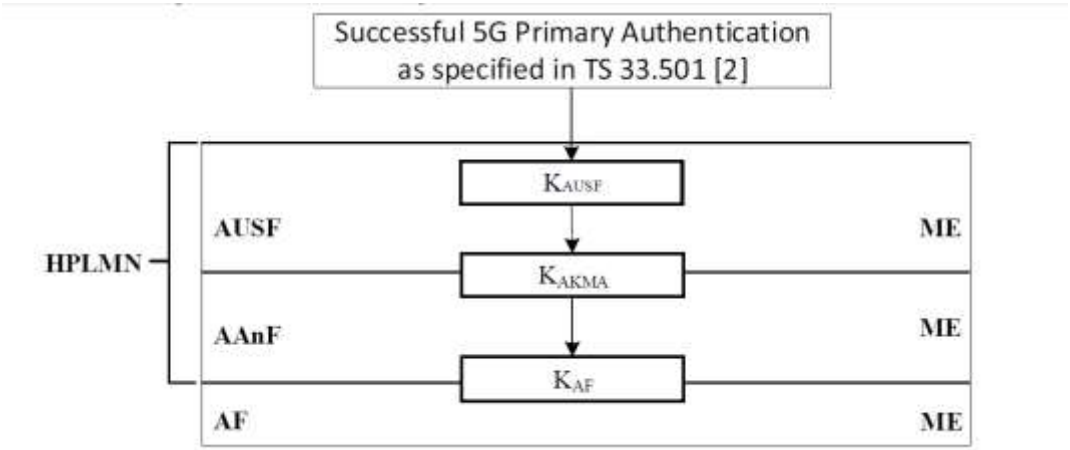
- AAnF – AKMA Anchor Function
  - New logical entity in the 5G Home Network that maintains UE's AKMA context(s) in the network & derives Application Function (AF) specific keys (K_{AF})
- Ua* is the application specific protocol that is secured using the KAF
  - Analogous to the Ua protocol in GBA
- Other AKMA relevant functions in 5G Core network: AUSF, UDM and NEF (if AF is located outside of operator's network)
  - UDM stores indication on whether AKMA service is available for a given UE. Also stores the address of AAnF(& AUSF) serving the UE
  - AUSF derives the AKMA Anchor Key (K_{AKMA}) after successful UE authentication and delivers it to AAnF

AKMA: Key Hierarchy



- If AKMA service is supported for a given UE, the AUSF is required to store the K_{AUSF}

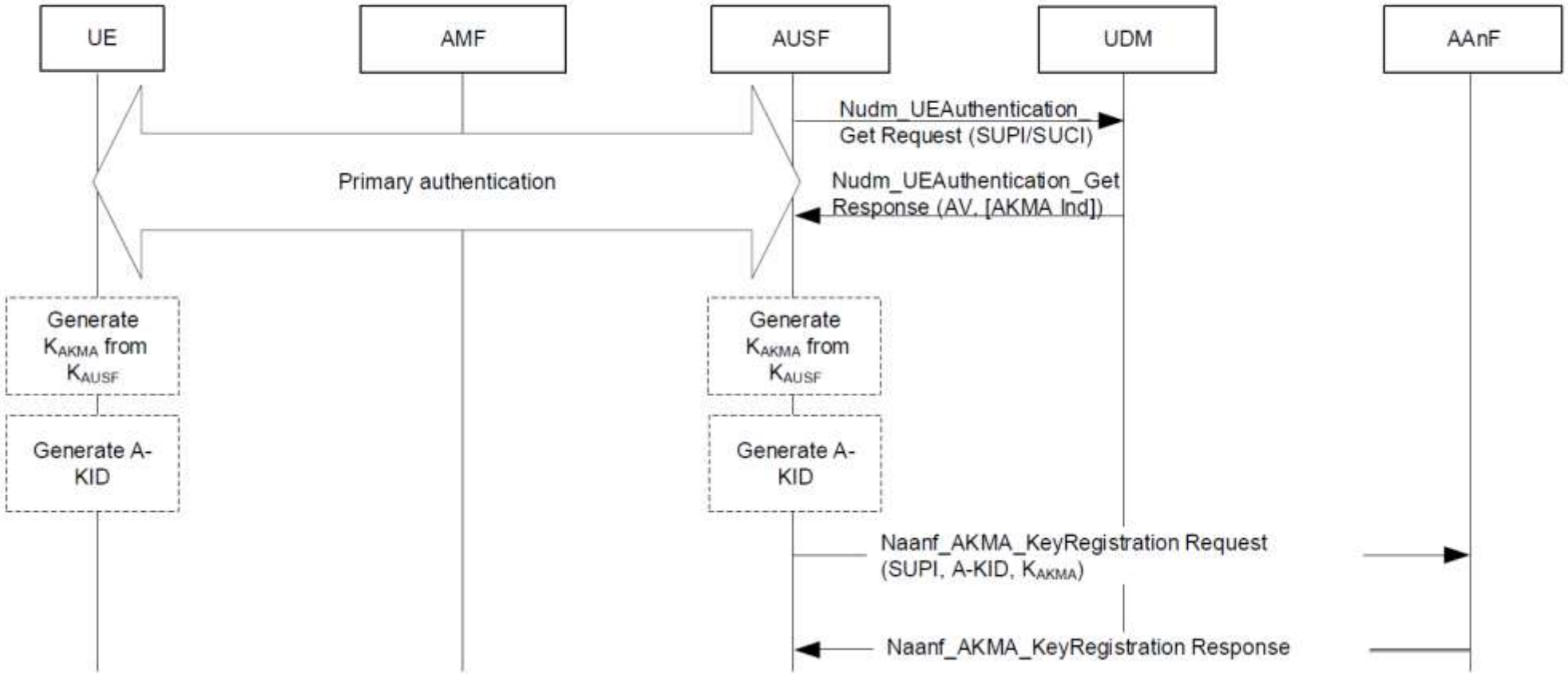
AKMA: Key Hierarchy (cont.)



- K_{AKMA} is the AKMA anchor key derived by AUSF and stored at AAnF
  - UE (ME) may derive KAKMA as needed (on-demand)
- K_{AF} is derived by AAnF; along with a lifetime, send to the AF
- K_{AKMA}, K_{AF} can be changed only by changing K_{AUSF}, i.e. by running 5G primary authentication

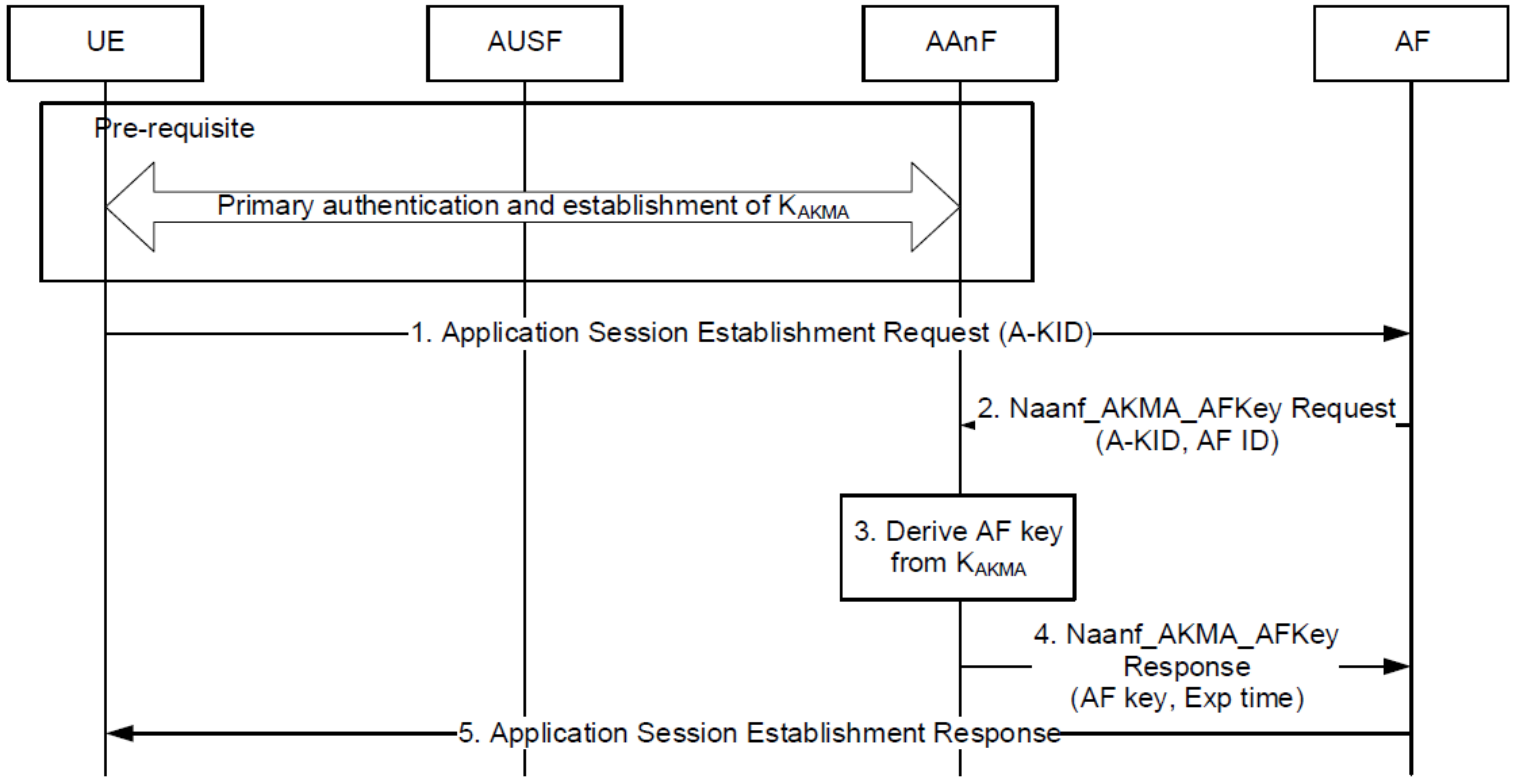
AKMA Procedures

Deriving AKMA Anchor Key - K_{AKMA}

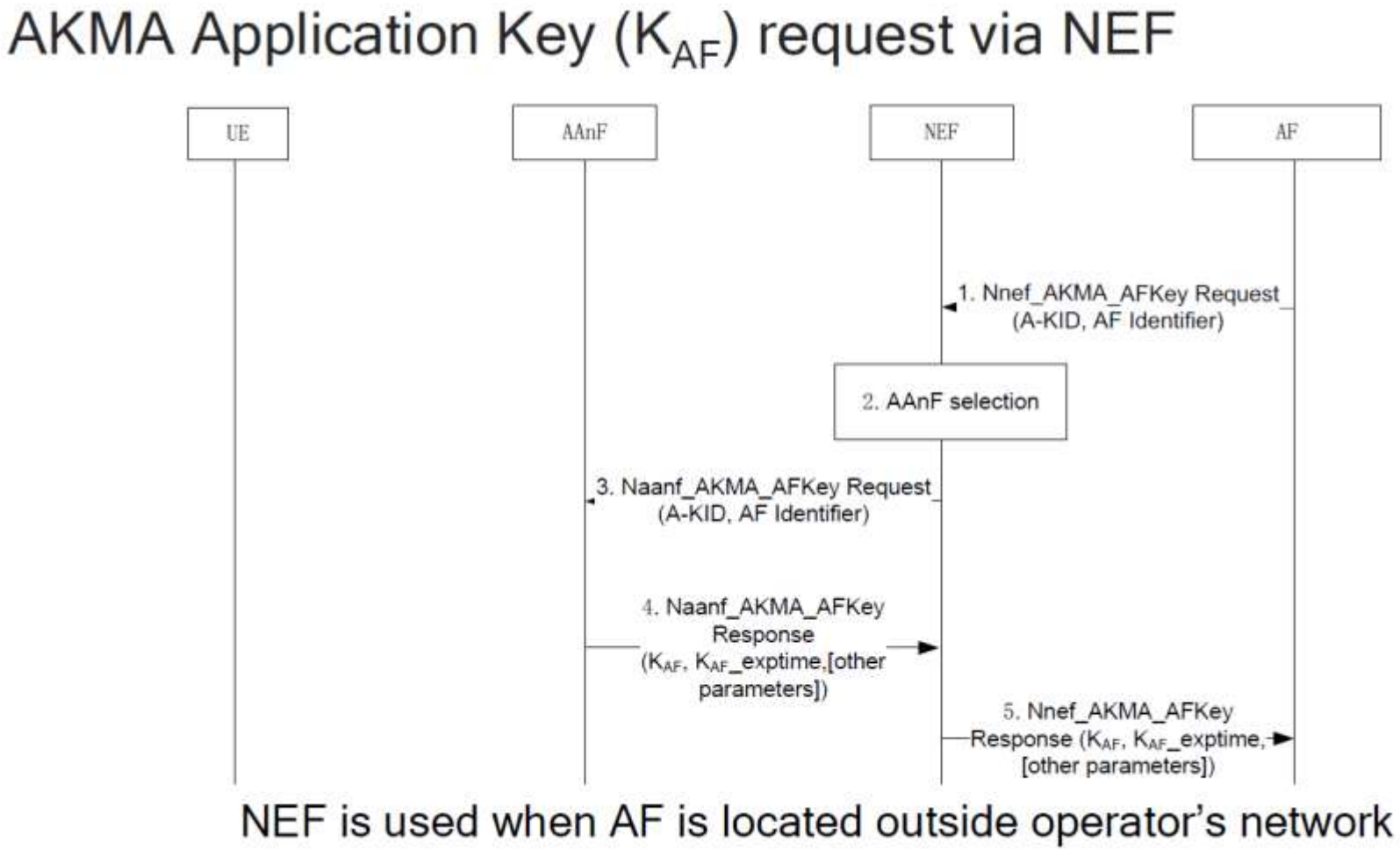


AKMA Procedures (cont.)

Deriving AKMA Application Key - K_{AF}



AKMA Procedures (cont.)



Qualcomm

Thank you

Follow us on:

For more information, visit us at:

www.qualcomm.com & www.qualcomm.com/blog