Background

- LAKE is about specifying a lightweight authenticated key exchange protocol for OSCORE (RFC 8613)

- The requirements for the lightweight AKE are based on the conditions for deploying OSCORE in constrained environments (RFC 7228)

- This is not a new subject in the IETF
  - On the agenda for ACE WG F2F meetings at IETF 96–99, 101–103
  - Extensively discussed in SecDispatch 2019, dedicated virtual interim March 5
  - BoF@IETF105
Requirements

- OSCORE Related
- Authentication
- Credentials
- Crypto Properties
- Application Data
- Lightweight
OSCORE Related

— At the end of the AKE the two parties shall agree on
  — OSCORE Master Secret with PFS and good amount of randomness
  — OSCORE Sender IDs of peer endpoint, arbitrarily short
  — COSE algorithms to use with OSCORE

— The AKE shall reuse CBOR, CoAP and COSE primitives and algorithms for low code complexity of a combined OSCORE and AKE implementation

— The AKE shall support the same transport as OSCORE, in particular CoAP.
— The AKE shall not duplicate functionality supported by the transport.
— The transport is assumed to handle:
  — packet loss, reordering, and duplication
  — message fragmentation
  — denial of service protection
The AKE shall support mutual authentication using PSK, RPK, and public key certificates
- Different public key credentials for different endpoints
  - e.g. certificates for the initiator and RPK for the responder
  - Support for different identification of credentials including key identifier, hash, certificate, URL

The AKE shall support identity protection
- public keys: against active attackers of one of the peers and against passive attackers of the other peer
  - symmetric keys: PSK identifier against active attackers

The AKE shall support negotiation of COSE crypto algorithms
- used with OSCORE (COSE AEAD algorithm and HMAC-based HKDF)
- used in the AKE (AEAD algorithm, KDF, signature algorithm, DH algorithm, ...)
  - Algorithm selection shall be protected against downgrade attacks
Compromise of the long-term keys shall not enable
- an attacker to compromise past session keys (Perfect Forward Secrecy)
- a passive attacker to compromise future session keys.

The AKE shall provide Key Compromise Impersonation (KCI) resistance.

The AKE shall protect against misbinding attacks and reflection attacks such as the Selfie attack.
Application Data

— The AKE shall support transport of Application Data to support a reduced total no. of round trips/no. of messages, and combined features, e.g. authorization together with authentication

— Example of Application Data:
  — Authorization information such as PoP Token, Authorization Voucher
  — Certificate Enrolment request, such as CSR

(Discussion of application data later in this meeting.)
Lightweight

- The AKE shall have as few round trips/messages as possible
- The messages shall be as small as reasonably achievable and fit into as few LoRaWAN packets and 6TiSCH frames as possible
- The amount of new code required on end systems which already have an OSCORE stack shall be as small as reasonably achievable
AKE Frequency

— Can we estimate how often we need to run the AKE/how many times during device lifetime?
— Not in general. Note that:

1. For some use cases, already one execution of the AKE is too heavy.
   — parallel executions of the AKE in a network formation loads down the network, or
   — the duty cycle makes the completion time too long for even one run of the protocol.

2. If a device reboots it may not be able to recover the security context, e.g. due to lack of persistent storage, and is required to establish a new security context for which an AKE is preferred. Reboot frequency may be difficult to predict in general.

3. To limit the impact of a key compromise, BSI, NIST and ANSSI and other organizations recommend frequent renewal of keys by means of a Diffie-Hellman key exchange.

Even if we are unable to give precise numbers, a lightweight AKE
— reduces the time for network formation and for AKE runs in challenging radio technologies
— allows devices to more quickly re-establish security in case of reboots, and
— allows us to support recommendations of frequent key renewal
Discussion Topics

☆ Static DH requirements
☆ Confidentiality protection of PSK identifier
☆ Security properties of application data
Static DH Requirements 1(2)

- Static DH keys shall be supported
- At least for RPK
- Significant improvement in overhead

<table>
<thead>
<tr>
<th></th>
<th>PSK (Sign)</th>
<th>RPK (ECDH)</th>
</tr>
</thead>
<tbody>
<tr>
<td>message_1</td>
<td>40</td>
<td>38</td>
</tr>
<tr>
<td>message_2</td>
<td>45</td>
<td>114</td>
</tr>
<tr>
<td>message_3</td>
<td>11</td>
<td>80</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>96</strong></td>
<td><strong>232</strong></td>
</tr>
</tbody>
</table>

Example: Message sizes with EDHOC-00

Party U

<table>
<thead>
<tr>
<th>TYPE, SUITES_U, G_X, C_U</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

message_1

<table>
<thead>
<tr>
<th>C_U, G_Y, C_V, AEAD( K_2; ID_CRED_V, AEAD(G_VX; CRED_V, TH_2) )</th>
</tr>
</thead>
</table>

message_2

<table>
<thead>
<tr>
<th>C_V, AEAD(K_3; ID_CRED_U, AEAD(G_UY; CRED_V, TH_2) )</th>
</tr>
</thead>
</table>

message_3

MAC instead of signature
Static DH Requirements 2(2)

— Both signature and static DH based authentication needs to be supported
  — Cannot assume static DH keys as the only type of public-key credentials
  — Common X.509 settings use public signature keys

— Support for mixed public key credentials
  — In terms of RPK / certificates (as mentioned previously)
  — Also in terms of static DH keys / public signature keys

\[
\begin{align*}
\text{Party U} & \quad \text{Type, Suites}_U, G_X, C_U \\
\hline
& \text{message}_1 \\
& C_U, G_Y, C_V, \text{AEAD}(K_2; \text{ID}_\text{CRED}_V, \text{Sig}(V; \text{CRED}_V, \text{TH}_2)) \\
\hline
& \text{message}_2 \\
& C_V, \text{AEAD}(K_3; \text{ID}_\text{CRED}_U, \text{AEAD}(G_{UY}; \text{CRED}_V, \text{TH}_2)) \\
\hline
\end{align*}
\]

\text{Or vice versa}

Confidentiality Protection of PSK Identifier

— ID-PSK may be encrypted in message 3
— Does not provide authentication of responder (party V)
— Adding a message conflicts with requirement "few messages"

Party U

<table>
<thead>
<tr>
<th>TYPE, SUITES_U, G_X, C_U, UAD_1</th>
<th>Party V</th>
</tr>
</thead>
<tbody>
<tr>
<td>message_1</td>
<td></td>
</tr>
<tr>
<td>C_U, G_Y, C_V, AEAD(K_2A; TH_2, UAD_2)</td>
<td></td>
</tr>
<tr>
<td>message_2</td>
<td></td>
</tr>
<tr>
<td>C_V, AEAD(K_3A; TH_3, ID_PSK), AEAD(K_3B; TH_3, PAD_3)</td>
<td></td>
</tr>
<tr>
<td>message_3</td>
<td></td>
</tr>
</tbody>
</table>
Identity Protection

— Sequence of desired goals where we may only be able to meet some level:
  — 0: all identifying information should be protected against passive network adversaries
  — 1: the identifying information of one device (say the initiator) must be protected from an active network attacker
  — 2: the identifying information of both devices must be protected from an active network attacker
  — 3: the identifying information of both devices must be deniable/repudiable, even if the peer is malicious

— Trade-offs
  — Identity protection of the symmetric protocol and authentication of responder/no. of messages
  — Disclosure of supported cipher suites vs. crypto agility
  — Connection ID could reveal information about the size of the server
Security Properties

- PFS against compromise of which key material
  - Loss of long-term key (initiator and/or responder)?
  - Loss of ephemeral key (initiator and/or responder)?
  - Bad RNG (initiator and/or responder)?

- Current assumption:
  - Protection against loss of long-term keys at the initiator and responder

- DISCUSS
  - Cost/benefit of protecting against loss of ephemeral key or bad RNG
Security Properties of Application Data

— Different requirements for application data (AD) in different messages:
  — AD1: unprotected
  — AD2: confidentiality/integrity protection against passive attacker
  — AD3: confidentiality/integrity protection

— AD must not violate AKE security properties
— Assumptions on AD shall be detailed by the specification

Client

RA/CA

Authz Service

AD1 = Authz Request

AD2 = Authz Token

AD3 = CSR

Certificate (reference)

AKE