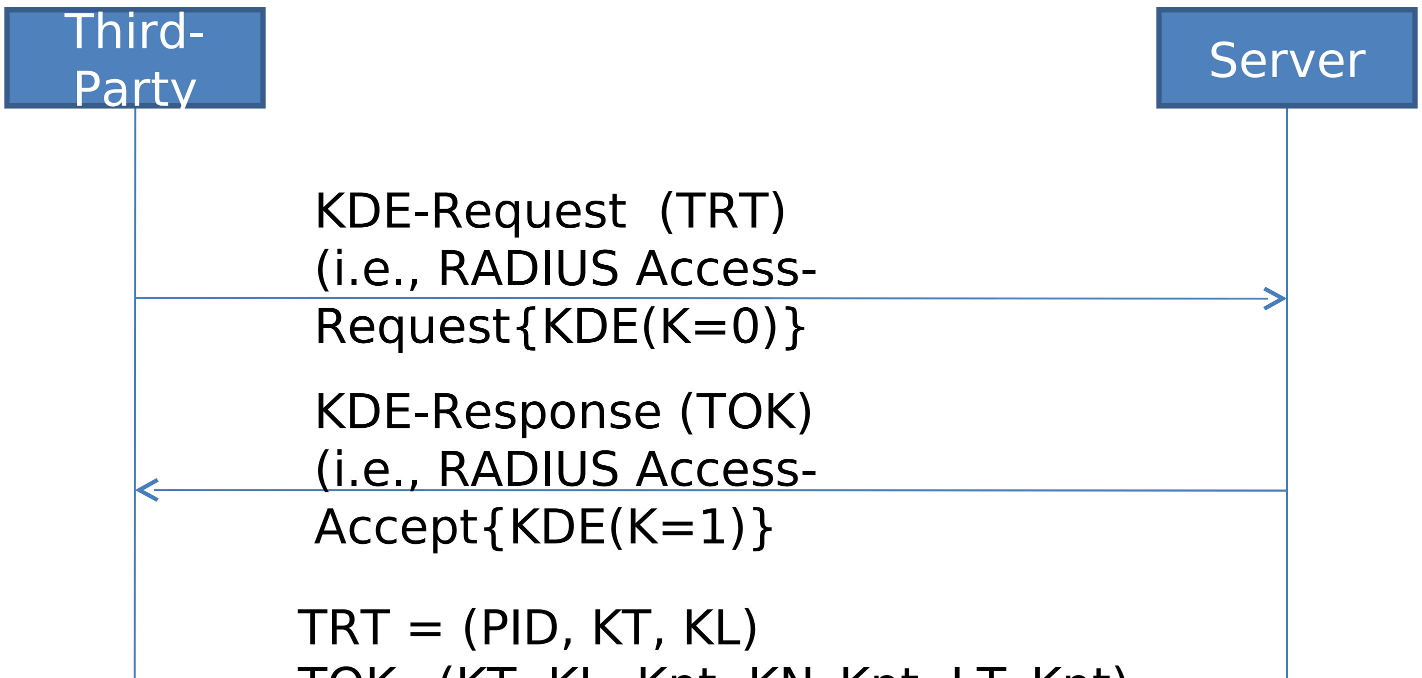
draft-ietf-hokey-key-mgm-04

Madjid Nakhjiri
Yoshihiro Ohba (Ed.)
Kedar Gaonkar
Lakshminath Dondeti
Vidya Narayanan
Glen Zorn

High-level Changes

- Merged with draft-gaonkar-radext-erp-attrs
- Put focus on distribution of USRK, DSRK and USDSRK over RADIUS
  - Relying on RADIUS security
- Removed “three-party” word
- Revised Security Considerations section

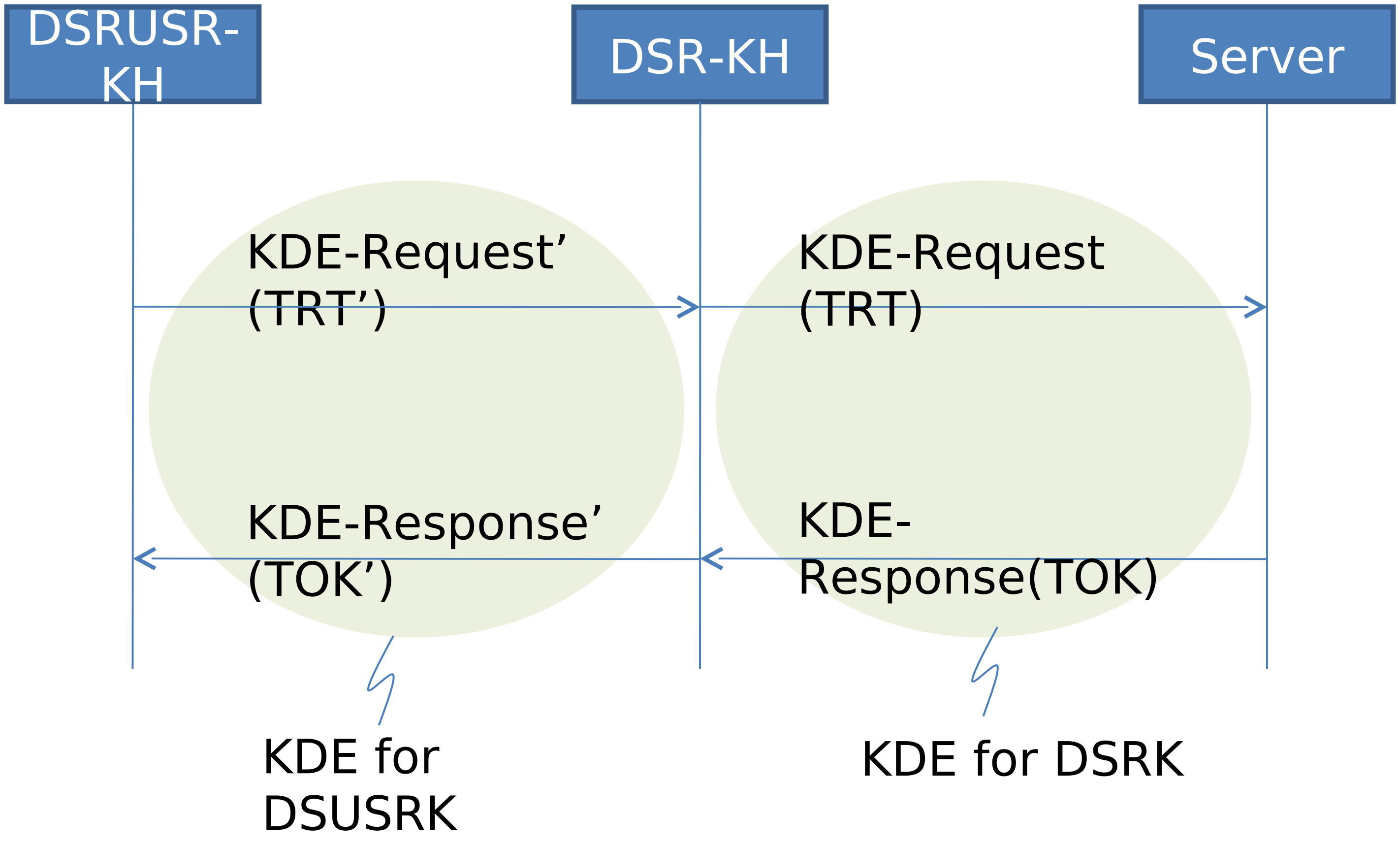
Basic Key Distribution Exchange (KDE) Sequence



TRT = (PID, KT, KL)
TOK=(KT, KL, Kpt, KN_Kpt, LT_Kpt)

PID: Peer ID, KT: Key Type, KL: Key Label
Kpt: USRK, DSRK or DSUSRK
KN_Kpt: Key Name, LT_Kpt: Key Lifetime

Combined KDE Sequence for distributing DSRK and DSUSRK



RADIUS KDE Attribute

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Type      |    Length     |K|  Reserved    |   Key Type    |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Key Label ...
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Key Name (included only when K=1) ...
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Key (included only when K=1) ...
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Key Lifetime (included only when K=1)                      |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

K=0 → KDE-Request
K=1 → KDE-Response
Key Type: 1 (DSRK), 2 (USRK), 3 (DSUSRK)
(See IANA Considerations section for detailed Key Type allocation policy)

When and how KDE Attr. is carried

- Explicit ERP Bootstrapping
  - KDE-Request is carried in a RADIUS Access-Request message that carries an EAP-Initiate message with the bootstrapping flag set
  - KDE-Response is carried in a RADIUS Access-Accept message that carries an EAP-Finish message with the bootstrapping flag set

- Implicit ERP bootstrapping
  - KDE-Request is included in the RADIUS Access-Request message that carries the first EAP-Response message from the peer
  - KDE-Response is carried in a RADIUS Access-Accept message that carries an EAP-Success

- In both cases, a value of the RADIUS User-Name attribute is used as the PID

Conflicting Messages (Prohibited patterns)

- Access-Accept/EAP-Message/EAP-Finish with 'R' flag set to 1
- Access-Reject/EAP-Message/EAP-Finish with 'R' flag set to 0
- Access-Reject/Keying-Material
- Access-Reject/KDE
- Access-Challenge/EAP-Message/EAP-Initiate
- Access-Challenge/EAP-Message/EAP-Finish
- Access-Challenge/KDE

Security Requirements on RADIUS Key Transport

- RADIUS messages that carry a KDE attribute MUST be encrypted and integrity and replay protected with a security association created by a RADIUS transport protocol such as TLS [I-D.ietf-radext-radsec].

- When there is an intermediary such as a RADIUS proxy on the path between the third-party and the server, there will be a series of hop-by-hop security associations along the path.

- The use of hop-by-hop security associations implies that the intermediary on each hop can access the distributed keying material.

- Hence the use of hop-by- hop security SHOULD be limited to an environment where an intermediary is trusted not to use the distributed key material.

Security Consideration on Lack of Peer Consent

- When a KDE-Request message is sent as a result of explicit ERP bootstrapping [RFC5296], cryptographic verification of peer consent on distributing a Kpt is provided by the integrity checksum of the EAP-Initiate message with the bootstrapping flag turned on.

- When a KDE-Request message is sent as a result of implicit ERP bootstrapping [RFC5296], cryptographic verification of peer consent on distributing a Kpt is not provided.
  - As a result, it is possible for a third-party to request a Kpt from the server and obtain the Kpt even if a peer actually does not support ERP, which can lead to an unintended use of a Kpt.