HOKEY 3-Party Key Distribution
(draft-ietf-hokey-key-mgm-01.txt)

Madjid Nakhjiri
Yoshihiro Ohba
Status

• Submitted -01 in November
• 14 issues are closed
• 2 issues are still open
**Key Distribution Model**

- **Peer (P)**
- **Server (S)**
- **Third Party (T)**

**Pre-existing Trust relationship / SA (Kps)**

**Kpt(=F(Kps))**

**Trust relationship / SA to be created (Kpt)**

**Pre-existing Trust relationship / SA (Kts)**

**Kpt** is used for dynamically establishing a trust relationship / SA between P and T
## Key Distribution Exchange

<table>
<thead>
<tr>
<th>Message Name (Parameters)</th>
<th>P</th>
<th>T</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>KDE0 (TID,SID,DID)</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>(TID, SID, DID) = (Third Party ID, Server ID, Domain ID)</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>KDE1 (PRT)</td>
<td></td>
<td>→</td>
<td></td>
</tr>
<tr>
<td>PRT(Peer Request Token) =</td>
<td></td>
<td>→</td>
<td></td>
</tr>
<tr>
<td>Int[KIps,(PID, TID, SID, DID, FVp, KT, KN_KIps)]</td>
<td></td>
<td>→</td>
<td></td>
</tr>
<tr>
<td>KDE2 (TRT)</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>TRT(Third Party Request Token ) =</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>Int[KIts, (PID, TID), PRT]</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>KDE3 (TOK)</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>TOK(Key Token) =</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>{PID, TID, KN_Kpt, KL_Kpt, Kpt, SAT}KCts</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>KDE4 (SAT)</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>SAT(Server Authorization Token) =</td>
<td></td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>Int[KIps,(PID, TID, SID, DID, FVp+1, KN_Kpt, KL_Kpt, KN_KIps)]</td>
<td></td>
<td>←</td>
<td></td>
</tr>
</tbody>
</table>

Int [K, X] : X || MIC(K,X)  
{X}K: X encrypted with K  
FVp: Freshness Value generated by P  
KT: Key Type  
KN_X : Key Name for key X  
KL_X: Key Lifetime for key X  
KIts (or IK): Key Integrity Key  
KCts (or CK): Key Encryption Key  
(IK and CK are derived from EMSK, USRK or DSUSRK depending on usage scenarios)
## Usage Scenarios

<table>
<thead>
<tr>
<th>Scenario #</th>
<th>Server</th>
<th>Third Party</th>
<th>Transported Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EAP Server</td>
<td>NAS</td>
<td>rMSK</td>
</tr>
<tr>
<td>2</td>
<td>EAP Server</td>
<td>USR-KH</td>
<td>USRK</td>
</tr>
<tr>
<td>3</td>
<td>EAP Server</td>
<td>DSR-KH</td>
<td>DSRK</td>
</tr>
<tr>
<td>4</td>
<td>DSR-KH</td>
<td>DSUSR-KH</td>
<td>DSUSRK</td>
</tr>
<tr>
<td>5</td>
<td>USR-KH</td>
<td>NAS</td>
<td>rMSK</td>
</tr>
<tr>
<td>6</td>
<td>DSUSR-KH</td>
<td>NAS</td>
<td>rMSK</td>
</tr>
<tr>
<td>7</td>
<td>USR-KH</td>
<td>USDSR-KH(*)</td>
<td>USDSRK</td>
</tr>
</tbody>
</table>

Note1: EAP Peer is always Client of 3-party key distribution

Note2: USDSR-KH is key holder for a domain-specific root key defined by each usage (and hence details are not defined in any HOKEY document)
Combined KDE

Peer

KDE0(TID,SID,DID)
KDE0’(TID’,SID’,DID’)
KDE1(PRT)
KDE1’(PRT’)
KDE4(SAT)
KDE4’(SAT’)

KDE2'(TRT’)
KDE3(TOK)

KDE1(PRT)
KDE2’(TRT’)
KDE4(SAT)
KDE3’(TOK’)

NAS

DSR-KH/

DSUSR-KH

EAP Server

12/06/2007
IETF70 HOKEY WG
Closed Issues (1/2)

- Issue 7 (replay attacks/nonce Np): -01 uses FV (freshness value) which allows time stamp or nonce. In the case of nonce, the draft has a warning that an additional mechanism may be required to assure freshness.

- Issue 8 (server id/domain id), -01 uses both server id and domain id to be more flexible.

- Issue 9 (carrying key names), -01 still carries key names to identity the latest key from older ones between a given pair of entities where each entity is still identified with PID, SID or TID.

- Issue 10 (carrying key types), -01 has now key type (KT) in message 1, requiring that the peer specifies the key type.

- Issue 11 (carrying DTID and DUID), -01 carries only TID for the third-party identity instead of DTID and DUID.

- Issue 12 (formatting of msg2, composition attack), the second Int[] is now carried inside the first Int[].

- Issue 13 (key length in message 3/4), key length is now integral part of key variable. Note KL_X now represents a key lifetime of key X instead of a key length of key X.
Closed Issues (2/2)

- Issue 14 (key name generation), -01 follows hokey-emsk draft for key name generation
- Issue 24 (editorial changes): Done
- Issue 25 (update figure 1 to match EMSK doc), Fig 1 has been updated to be consistent with hokey-emsk doc
- Issue 26 (references to HOKEY/HRK/etc), HRK and DSHRK are removed
- Issue 29 (hierarchy depth, DSUSRK children): -01 has only one usage for a child key of DSUSRK, that is ERX usage for rMSK derived from DSUSRK
- Issue 30 (terminology for DSRK child keys): KX and KY are removed
- Issue 31 (remove section 5.1), Section 5.1 is removed (except for CK and IK)
Open Issue: Issue 27
(Protocol Format)

- Formal protocol format specification will be added in the next revision
- But the format should be generic enough to be carried in various transport protocols
Open Issue: Issue 28

• -01 still mandate key encryption between server and 3rd party. Instead, the following note has been added in Security Considerations section:

"EDITOR'S NOTE: For a key distribution mechanism that works with indirect trust relationship, a Kerberos-like key distribution protocol that supports "inter-realm" keys would be needed."

• Should we allow hop-by-hop encryption?