OAKLEY Key Determination Protocol

author: Hilarie Orman
Computer Science Department
University of Arizona
May 1996
draft-ietf-ipsec-oakley-01.txt
OAKLEY: A *Family of Generic Protocols* for Key Distribution

1. **Shared Key:** common keying information state
   - key name, key material
   - identities of the (two) parties
   - selected, agreed-upon algorithms (i.e., encryption, hashing, authentication.)

2. **Key Distribution:** key generation (e.g., by Diffie-Hellman modular exponentiation) as opposed to key transport (i.e., key wrapping in other keys)

3. **Generic protocols:** flexible selection of mutually agreeable algorithms for
   - encryption (e.g., DES, IDEA)
     - group representation (e.g., MODP, ECP, EC2N)
   - hashing (e.g., MD5, SHA)
   - authentication (e.g., RSA, DSS)

4. **Family of Protocols:** flexible selection of protocol
   - modes:
     - aggressive, conservative, main, quick, new_group
   - services:
     - anti-clogging (i.e., stateless cookie exchange)
     - identity authentication and non-repudiation (via use of digital signatures)
     - Perfect Forward Secrecy and Back Traffic Protection for keys
     - identity privacy
     - Perfect Forward Secrecy for private identities
OAKLEY: Protocol Properties

1. Clogging (Denial of Service) Avoidance
   - key generation by modular exponentiation => intensive computation
     => possibility of clogging (denial of service) attacks
   - avoidance measure: use of “cookies”
     - (stateless) cookie
     - recipient’s net (IP) address and port no.
     - sender’s net (IP) address and port no.
     - secret local value of owner (that expires periodically)
     - cookie distribution to others => willingness to perform
       intensive computation upon return of own cookie
     - must check receipt of own cookie against sender’s net. address

2. Authentication is independent of key generation
   - authentication validates binding <identity, keying material>
   - authentication performed before key generation
   - authentication performed by a different method (e.g., use of public-key certificates)

3. Key generation - keying material:
   - Diffie-Hellman exponentials (with default or privately shared groups)
   - secret, high-entropy values
   - pre-distributed keys

4. Assumed Certificate Distribution and Validation Policies
OAKLEY: Examples of Protocol Rules (Conventions)

Message Parsing Rules
- “left-to-right parsing
- null field = multiprecision integer zero

Protocol Negotiation Rules
- Cookie rules:
  o CKY-I = 0, CKY-R = 0 => cookie request / conservative prot.
  o CKY-I = 0, CKY-R /= 0 => cookie response / conservative prot.
  o CKY-I /= 0, CKY-R = 0 => aggressive, main prot.
- Null values returned by responder, w/ subsequent fields null => unacceptable offer
- GRP = null, g^x field = null => unacceptable accept first proposal in EHAO

Service Negotiation Rules
- GRP = null, nonce in g^x field => accept first proposal for no PFS for Keys and IDs
- GRP, g^x, IDs = non-null => request PFS for Keys but not IDs
- GRP, g^x = non-null, IDs = null => request PFS for Keys and IDs

State Transition Rules
- timeout / retransmission
- cookie verification
- signature, prf verification

Need: a specification language for families of generic protocols
Aggressive Example 1: PFS for Keys, Public IDs, Digital Signatures

**Initiator**

CKY-I, 0, OK_KEYX, GRP, $g^x$, EHAO, NIDP, ID(I), ID(R), $N_i$, 0, $S \{ \text{ID(I)} | \text{ID(R)} | N_i | 0 | \text{GRP} | g^x | \text{EHAO} \} K_i$

**Responder**

CKY-R, CKY-I, OK_KEYX, GRP, $g^y$, EHAS, NIDP, ID(R), ID(I), $N_r$, $N_i$, $S \{ \text{ID(R)} | \text{ID(I)} | N_r | N_i | \text{GRP} | g^y | \text{EHAS} \} K_r$

CKY-I, CKY-R, OK_KEYX, GRP, $g^x$, EHAO, NIDP, ID(I), ID(R), $N_i$, $N_r$, $S \{ \text{ID(I)} | \text{ID(R)} | N_i | N_r | \text{GRP} | g^x | \text{EHAO} \} K_i$

**KEYID** = CKY-I | CKY-R  
**sKEYID** = prf ($N_i$, $N_r$, $g^{xy}$ | CKY-I | CKY-R)

**Notes: State Transition**
- first choices: H, A => state signature verification (certificates not shown)
- cookie verification (e.g., Initiator: CKY-I vs. net. address of incoming message)
- weak liveness check: ability to repeat other party’s cookie in order
Main Mode (ISAKMP: Identity Protection Exchange)

Initiator

| CKY-I, 0, OK_KEYX, GRP, EHAO |

Responder

| CKY-R, CKY-I, OK_KEYX, GRP, EHAS |

| CKY-I, CKY-R, OK_KEYX, GRP, g^x, EHAO, NIDP, N_i, 0, |

| CKY-R, CKY-I, OK_KEYX, GRP, g^y, EHAS, NIDP, N_r, N_i, |

| CKY-I, CKY-R, OK_KEYX, GRP, g^x, EHAO, IDP*, ID(I), 0, |

| S { CKY-I | CKY-R | ID(I) | N_i | N_r | GRP | g^{xy} | EHAO } K_i |

| CKY-R, CKY-I, OK_KEYX, GRP, g^y, EHAS, IDP*, 0, ID(R), |

| S { CKY-R | CKY-I | ID(R) | N_r | N_i | GRP | g^{xy} | EHAAS } K_r |

where * key = prf {0, g^{xy}}

KEYID = CKY-I | CKY-R  
sKEYID = prf (N_i | N_r , g^{xy} | CKY-I | CKY-R )
Aggressive Example 2: PFS for Keys, Private IDs via Proxy R’

**Initiator**

\[
\text{CKY-I} , \ 0 , \ OK \ \text{KEYX} , \ \text{GRP} , \ g^x , \ \text{EHAO} , \ \text{NIDP} , \ 0 , \ \text{ID(R')} , \\
\text{E} \ \{ \ \text{ID(I)} , \ \text{ID(R)} , \ \text{E} \ \{ \ N_i \} \ K_r \} \ K_r .
\]

**Responder**

\[
\text{CKY-R} , \ \text{CKY-I} , \ \text{OK} \ \text{KEYX} , \ \text{GRP} , \ g^y , \ \text{EHAS} , \ \text{NIDP} , \ \text{ID(R')} , \ 0 , \\
\text{E} \ \{ \ \text{ID(R)} , \ \text{ID(I)} , \ N_r \} \ K_i , \ \text{prf} \ \{ \ \text{Kir} , \ \text{ID(R)} | \ \text{ID(I)} | \ N_r | \ N_i | \ \text{GRP} | \ g^y | \ g^x \} \\
\text{where} \ \text{Kir} = \ \text{prf} \ \{ \ 0 , \ N_i | N_r \}
\]

\[
\text{CKY-I} , \ \text{CKY-R} , \ \text{OK} \ \text{KEYX} , \ \text{GRP} , \ 0 , \ 0 , \ \text{NIDP} , \ 0 , \ \text{ID(R')} , \\
\ \text{prf} \ \{ \ \text{Kir} , \ \text{ID(I)} | \ \text{ID(R)} | \ N_i | N_r | \ \text{GRP} | g^x | \ g^y \}
\]

**KEYID** = \text{CKY-I} | \text{CKY-R} \quad \text{sKEYID} = \ \text{prf} \ \{ \ N_i | N_r , \ g^{xy} | \ \text{CKY-I} | \ \text{CKY-R} \}

Notes: State Transition (proxy R’ maintains state)
- first choices: E, H (prf); certificates not shown
- cookie verification (e.g., Initiator: CKY-I vs. net. address of incoming message)
- weak liveness check: ability to repeat other party’s cookie in order
Aggressive Example 3: Private IDs and W/O Diffie-Hellman

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKY-I, 0, OK KEYX, 0, 0, EHAO, NIDP, 0, ID(R'), E { ID(I), ID(R), s Ki } Kr', prf { Kir, ID(R) \mid ID(I) }</td>
<td>CKY-R, CKY-I, OK KEYX, 0, 0, EHAS, NIDP, ID(R'), 0, E { ID(R), ID(I), sKr } K_r, prf { Kir, ID(R) \mid ID(I) \mid sK_r \mid sK_i }</td>
</tr>
</tbody>
</table>

where Kir = prf \{ 0, sK_i \mid sK_r \}

CKY-I, CKY-R, OK KEYX, 0, 0, EHAS, NIDP, 0, ID(R'), prf \{ Kir, ID(I) \mid ID(R) \mid sK_i \mid sK_r \}

KEYID = CKY-I \mid CKY-R
sKEYID = prf ( K_{ir}, CKY-I \mid CKY-R )

Notes: State Transition (proxy R' maintains state); replay detection? sKEYID expiration
- first choices: E, H (prf); certificates not shown
- cookie verification (e.g., Initiator: CKY-I vs. net. address of incoming message)
- weak liveness check: ability to repeat other party’s cookie in order
Conservative Example: PFS for Keys and for Private IDs

Initiator

\[ 0, 0, \text{OK KEYX} \]
\[ \rightarrow \]
\[ 0, \text{CKY-R, OK KEYX} \]

\[ \text{CKY-I, 0, OK KEYX, GRP, g}^x, \text{EHAO} \]

\[ \leftarrow \]
\[ \text{CKY-R, CKY-I, OK KEYX, GRP, g}^y, \text{EHAS} \]

\[ \text{CKY-I, CKY-R, OK KEYX, GRP, g}^x, \text{IDP*}, \quad \begin{array}{c} \text{ID(I)} \text{, ID(R)} \text{, E}\{N_i\}K_r \end{array} \]

where * key = prf \{ 0, g^{xy} \}

\[ \leftarrow \]
\[ \text{CKY-R, CKY-I, OK KEYX, GRP, 0, 0, IDP*}, \quad \begin{array}{c} \text{E}\{N_r, N_i\}K_i, \text{ID(R)}, \text{ID(I)} \end{array} \]

\[ \text{prf} \{ N_r | N_i, \text{GRP} | g^{xy} | \text{ID(R)}, \text{ID(I)} \} \]

\[ \leftarrow \]
\[ \text{CKY-I, CKY-R, OK KEYX, GRP, 0, 0, IDP*}, \quad \begin{array}{c} \text{prf} \{ N_i | N_r, \text{GRP} | g^{xy} | \text{ID(I)}, \text{ID(R)} \} \end{array} \]

\[ \text{KEYID} = \text{CKY-I} | \text{CKY-R} \quad \text{sKEYID} = \text{prf} (N_i | N_r, g^{xy} | \text{CKY-I} | \text{CKY-R}) \]

Notes: State Transition

- first choices: E, H (prf) ; certificates not shown
- cookie verification (e.g., Initiator: CKY-I vs. net. address of incoming message)
- weak liveness check: ability to repeat other party’s cookie in order
Quick_Mode Example

Initiator

\[ \text{KEYID, INEWKEYRQ, } N_i, \text{prf ( sKEYID, } N_i) \]

Responder

\[ \text{KEYID, INEWKEYRS, } N_r, \text{prf ( sKEYID, } 1 | N_r | N_i) \]

\[ \text{KEYID, INEWKEYRP, 0, prf ( sKEYID, } 0 | N_i | N_r) \]

\[ \text{NKEYID = } N_i | N_r \quad \text{sKEYID = prf ( sKEYID, } N_i | N_r) \]

Note: Extensive use of QUICK_MODE consumes the entropy of \( g^{xy} \) and, hence, QUICK MODE should be used sparingly.
External_Key Example

Initiator

\[ \text{KEYID}, \ I\text{EXTKEY}, N_1, \text{prf}(s\text{KEYID}, N_1) \]

\[ \text{KEYID}, \ I\text{EXTKEY}, N_r, \text{prf}(s\text{KEYID}, 1 | N_r | N_i) \]

\[ \text{KEYID}, \ I\text{EXTKEY}, K_{ir} \ xor \ s\text{NEWKEYID}^*, \text{prf}(K_{ir}, s\text{NEWKEYID} | N_i | N_r) \]

where \( K_{ir} = \text{prf}(s\text{KEYID}, | N_i | N_r) \)

Resonder

\[ \text{NKEYID} = \text{prf}(s\text{KEYID}, 1 | N_i | N_r) \quad \text{sNKEYID} = K_{ir} \ xor (K_{ir} \ xor s\text{NEWKEYID}) \]

Note: \( \text{length}(s\text{KeyID}) \geq \text{length}(s\text{NEWKEYID}) \)
New_Group Selection

Initiator

KEYID, INEWGRP, Desc(New_Group), Na,
prf(sKEYID, Desc(New_Group) | Na)

Responder

KEYID, INEWGRPRS, Nb, Na,
prf(sKEYID, Nb | Na, Desc(New_Group))

KEYID, INEWGRPACK,
prf(sKEYID, Na | Nb, Desc(New_Group))

Notes:

- Na, Nb =/= Ni, Nr or nonces used for current GRPID
- Desc(G) = group descriptor
- GRPID <-> Desc(NEWGRP) must be stored anew (with new IDs in KEYID)
Group Descriptors - 2 Examples

Group Type: MODP /* modular exponentiation group, mod P*/
Size of Field (in bits): $\lceil \log_2 P \rceil$ a 32-bit integer
Defining Prime P: a multi-precision integer
Generator G: a multi-precision integer $\ 2 \leq G \leq P-2$
optional:
Largest prime factor of P-1 : the multiprecision integer Q
Strength of Group: a 32-bit integer (approx. the no. of key bits protected; $\log_2$ of workfactor)

Group Type: ECP /* elliptic curve group, mod P */
Size of Field (in bits): $\lceil \log_2 P \rceil$ a 32-bit integer
Defining Prime P: a multi-precision integer
Generator (X, Y): two multi-precision integers $(X, Y \leq P)$
Parameters of the curve A, B: two multi-precision integers $(A, B \leq P)$
optional:
Largest prime factor of group order : the multi-precision integer
Order of the group: a multi-precision integer
Strength of Group: a 32-bit integer (approx. the no. of key bits protected; $\log_2$ of workfactor)

elliptic curve equation: $Y^2 = X^3 + AX + B$