EKE2 (with mutual authentication)

Protocol Purpose

Encrypted key exchange with mutual authentication

Definition Reference

http://citeseer.ist.psu.edu/bellare00authenticated.html

Model Authors

- Haykal Tej, Siemens CT IC 3, 2003
- Sebastian Mödersheim, ETH Zürich, December 2003

Alice&Bob style

```
1. A -> B : A.{exp(g,X)}_K(A,B)

    B computes master key MK
    MK = H(A,B,exp(g,X),exp(g,Y),exp(g,XY))

2. B -> A : {exp(g,Y)}_K(A,B), H(MK,1)

     A computes master key MK

3. A -> B : H(MK,2)

     Session key K = H(MK,0)

H : hash function
K(A,B): password (shared key)
```

Model Limitations

None

Problems considered: 3

Attacks Found

None

Further Notes

For information, this protocol is an example of the proposition done in http://citeseer.ist.psu.edu/bellare00authenticated.html showing that any secure AKE (Authentication Key Exchange) protocol can be easily improved to also provide MA (Mutual Authentication).

HLPSL Specification

```
role eke2_Init (A,B : agent,
                G: text,
                H: function,
                Kab : symmetric_key,
                Snd,Rcv: channel(dy))
played_by A
def=

  local State     : nat,
        X         : text,
        GY        : message,
        MK_A,MK_B : message

  const two : text,
        sec_i_MK_A : protocol_id

  init  State := 0

  transition

   1. State = 0  /\ Rcv(start) =|>
      State'= 1 /\ X' := new()
```

```
                /\ Snd(A.{exp(G,X')}_Kab)

   2. State = 1  /\ Rcv({GY'}_Kab.H(H(A.B.exp(G,X).GY'.exp(GY',X)).one)) =|>
      State':= 2 /\ MK_A' := A.B.exp(G,X).GY'.exp(GY',X)
                 /\ MK_B' := MK_A'
                 /\ Snd(H(H(MK_A').two))
                 /\ secret(MK_A',sec_i_MK_A,{A,B})
                 /\ request(A,B,mk_a,MK_A')
                 /\ witness(A,B,mk_b,MK_B')
end role
```

```
role eke2_Resp (B,A : agent,
                G: text,
                H: function,
                Kab : symmetric_key,
                Snd,Rcv : channel(dy))
played_by B
def=

  local State     : nat,
        Y         : text,
        GX        : message,
        MK_A,MK_B : message

  const one : text,
        sec_r_MK_B : protocol_id

  init  State := 0

  transition

   1. State = 0  /\ Rcv(A.{GX'}_Kab) =|>
      State':= 1 /\ Y' := new()
                 /\ MK_B' := A.B.GX'.exp(G,Y').exp(GX',Y')
                 /\ MK_A' := MK_B'
                 /\ Snd({exp(G,Y')}_Kab.H(H(MK_B').one))
                 /\ secret(MK_B',sec_r_MK_B,{A,B})
                 /\ witness(B,A,mk_a,MK_A')
```

```
   2. State = 1  /\ Rcv(H(H(MK_B).two)) =|>
      State':= 2 /\ request(B,A,mk_b,MK_B)

end role
```

```
role session (A,B: agent,
              G: text,
              H: function,
              Kab: symmetric_key) def=

   local   SA,RA,SB,RB: channel(dy)

   composition

           eke2_Init(A,B,G,H,Kab,SA,RB)  /\
           eke2_Resp(B,A,G,H,Kab,SB,RA)

end role
```

```
role environment() def=

  const mk_a, mk_b  : protocol_id,
        a,b,c       : agent,
        kab,kai,kib : symmetric_key,
        g           : text,
        h           : function

  intruder_knowledge  = {a,b,c,kai,kib}

  composition

        session(a,b,g,h,kab) /\
        session(a,i,g,h,kai) /\
        session(i,b,g,h,kib)

end role
```

```
goal

  %secrecy_of MK
  secrecy_of sec_i_MK_A, sec_r_MK_B

  %Eke2_Init authenticates Eke2_Resp on mk_a
  authentication_on mk_a
  %Eke2_Resp authenticates Eke2_Init on mk_b
  authentication_on mk_b

end goal
```

```
environment()
```

References