UMTS-AKA

Protocol Purpose

Authentication and Key Agreement

Definition Reference

`http://www.3gpp.org/ftp/tsg_sa/WG3_Security/_Specs/33902-310.pdf`

Model Authors

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

Alice&Bob style

S is the server, M is the mobile set, they share a secret key k(M).

Both S and M have an own version of a sequence number, that they try to maintain synchonized.

Using k(M), a random number (nonce) r, his sequence number seq, when S receives a request from M (or whenever he wishes this part is not modelled here), S generates:

```
   res = F2(k(M); r)   where F2 hash
   CK =  F3(k(M); r)   where F3 one-way
   IK =  F4(k(M); r)   where F4 one-way
   Ka =  F5(k(M); r)   where F5 one-way
   AUTN = {seq}Ka; F1(k(M); seq; r)  where F1 hash

 M -> S : M
 S -> M : r; {seq}_Ka; F1(k(M); seq; r)

      from r M calculates KA, then seq, then checks if F1(k(M); seq; r) OK
      if yes, M increments his seq number and responds:

 M -> S : F2(k(M); r)
```

The goal is that at the end both authenticate each other and share the value of `CK` and `IK`.

Problems considered: 3

Attacks Found

None

HLPSL Specification

```
role server(S,M : agent,
            Snd, Rec: channel(dy),
            K_M: symmetric_key,
            Seq : text,
            F1,F2,F5: function)
played_by S
def=

  local State : nat,
        R     : text

  const r1,r2,sseq1 : protocol_id,
        add         : function

  init  State := 1

  transition

    1.  State   = 1 /\ Rec(M)
        =|>
        State' := 2 /\ R' := new()
                    /\ Snd(R'.{Seq}_F5(K_M.R').F1(K_M.Seq.R'))
                    /\ secret(Seq,sseq1,{S,M})
                    /\ witness(S,M,r1,R')

    2.  State   = 2 /\ Rec(F2(K_M.R))
        =|>
        State' := 3 /\ Seq' := add(Seq,1)
                    /\ wrequest(S,M,r2,R)

end role
```

```
role mobile(M,S:agent,
            Snd, Rec: channel(dy),
            K_M: symmetric_key,
            Seq: text,
            F1,F2,F5: function)
played_by M
def=

  local State :nat,
        R     :text

  const
        r1,r2,sseq2 : protocol_id

  init  State := 1

  transition

    1.  State = 1 /\ Rec(start) =|>
        State'= 2 /\ Snd(M)

    2.  State = 2 /\ Rec(R'.{Seq}_F5(K_M.R').F1(K_M.Seq.R')) =|>
        State'= 3 /\ Snd(F2(K_M. R'))
                  /\ secret(Seq,sseq2,{M,S})
                  /\ wrequest(M,S,r1,R')
                  /\ witness(M,S,r2,R')

end role
```

```
role session(M,S: agent,
             K_M: symmetric_key,
             Seq: text,
             F1,F2,F5: function,
             SA,RA,SB,RB: channel(dy)) def=

   composition

         mobile(M,S,SA,RA,K_M,Seq,F1,F2,F5)
      /\ server(S,M,SB,RB,K_M,Seq,F1,F2,F5)

end role
```

```
role environment() def=

 local Sa1,Ra1,Ss1,Rs1 : channel (dy)

 const r1, r2                : protocol_id,
       a, i, s               : agent,
       k_as, k_is, kai       : symmetric_key,
       f1, f2, f5            : function,
       seq_as, seq_is, seq_ai : text

 intruder_knowledge={a,s,i,f1,f2,f5}

 composition

        session(a,s,k_as,seq_as,f1,f2,f5,Sa1,Ra1,Ss1,Rs1)
% /\    session(i,s,k_is,seq_is,f1,f2,f5,si1,ri1,ss2,rs2)
% /\    session(a,i,k_ai,seq_ai,f1,f2,f5,sa2,ra2,si2,ri2)

end role
```

```
goal

  secrecy_of sseq1,sseq2
  %Mobile weakly authenticates Server on r1  % the nonce R
  authentication_on r1
  %Server weakly authenticates Mobile on r2  % the nonce R
  authentication_on r2
```

```
end goal
```

```
environment()
```

References