We ignore length field (as it cannot be, yet, expressed in HLPSL), use fresh nonce to model RDM, and assume 'DelayedAuthReq' token is enough to specify alrithm, type of auth, and type of RDM.

The server returns the nonce + 1 (or succ(nonce) to be exact) instead of a timestamp with a higher value.

Problems considered: 2

```
  init State := 0

  transition

   1. State = 0
      /\ Rcv(C.delayedAuthReq.Time1')
      =|>
      State' := 1
      /\ Sig' := H(S,delayedAuthReq,succ(Time1'),K)
      /\ Snd(S.delayedAuthReq.succ(Time1').KeyID(K).Sig')
      /\ witness(S,C,sig,Sig')

end role
```

```
role session(C, S           : agent,
             H, KeyID       : function,
             K              : text)
def=

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

  composition
        dhcp_Delayed_Server(S,C,H,KeyID,K,SA,RA) /\
        dhcp_Delayed_Client(C,S,H,KeyID,K,SB,RB)

end role
```

```
role environment()
def=

 const a, b       : agent,
```

```
                        k3}

 composition
        session(a,b,h,keyid,k1)
    /\  session(a,i,h,keyid,k2)
    /\  session(i,b,h,keyid,k3)

end role
```

```
goal
  secrecy_of sec_k

  %DHCP_Delayed_Client authenticates DHCP_Delayed_Server on sig
  authentication_on sig
end goal
```

```
environment()
```

References