```
%Formal Verification of RLAS
role nodeU (U,V: agent,
            Hash: hash_func,
             Qca: public_key,
  Key1,Key2,Ks1: symmetric_key,
       SND, RCV: channel (dy))
played_by U def=
local
State                                :nat,
Idu,Certu,Lt,Idca,Nv,Certv,Nu,Idv  :text,
Token1,Token2,Z1,Z2                  :message

init State:= 0
transition
1. State = 0  /\ RCV(start)  =|>
  State':= 1  /\ Nu' := new()
              /\ Key1' := xor(Nu,xor(Idu,Idca))
              /\ Token1' := Hash(Certu.Lt.Nu.Idu.Idca)
              /\ Z1' := {Certu.Lt.Nu.Idu.Idca}_Ks1
              /\ SND(Token1',Z1')
              /\ secret ({Certu,Nu'},sub1,{U,V})


2. State = 2  /\ RCV(Token2',Z2') =|>
   State':= 3 /\ Key2' := xor(Nv,xor(Idv,Idca))
              /\ Token2' := Hash(Certv.Lt.Nv.Idv.Idca)
              /\ Z2' := {Certv.Lt.Nv.Idv.Idca}_Key2'
              /\ witness(U,V,nodeV_nodeU_lt,Lt)
end role

role nodeV (U,V: agent,
            Hash: hash_func,
             Qca: public_key,
  Key1,Key2,Ks1: symmetric_key,
       SND, RCV: channel (dy))
played_by V def=
local
State                                :nat,
Idu,Certu,Lt,Idca,Nv,Nu,Certv,Idv,E:text,
Token1,Token2,Z1,Z2                  :message

init State:= 1
transition
1. State = 1  /\ RCV(Token1',Z1')  =|>
   State':= 2 /\ Z1' := {Certu.Lt.Nu.Idu.Idca}_Ks1
              /\ Key1' := xor(Nu,xor(Idu,Idca))
              /\ Token1' := Hash(Certu.Lt.Nu.Idu.Idca)
              /\ Key2' := xor(Nv,xor(Idv,Idca))
              /\ Token2' := Hash(Certv.Lt.Nv.Idv.Idca)
              /\ Z2' := {Certv.Lt.Nv.Idv.Idca}_Key2'
              /\ SND (Token2',Z2')
              /\ secret ({Certv,Nv},sub2,{U,V})
              /\ witness(V,U,nodeU_nodeV_lt,Lt)
end role

role session (U,V: agent,
             Hash: hash_func,
              Qca: public_key,
    Key1,Key2,Ks1: symmetric_key)
def=
local SU,RU,SV,RV: channel(dy)
composition
   nodeU(U,V,Hash,Qca,Key1,Key2,Ks1,SU,RU)
/\ nodeV(U,V,Hash,Qca,Key1,Key2,Ks1,SV,RV)
end role

role environment ()
```

```
def=
const nodeU,nodeV: agent,
qca: public_key,
key1,key2,ks1,key1i,key2i,ks1i: symmetric_key,
idu,certu,lt,idca,e,nv,nu,certv,idv: text,
h: hash_func,
nodeU_nodeV_lt,nodeV_nodeU_lt,sub1,sub2: protocol_id

intruder_knowledge={nodeU,nodeV,h,key1i,key2i,ks1i,qca}

composition
session(nodeU,nodeV,h,qca,key1,key2,ks1)
/\session(nodeU,i,h,qca,key1i,key2i,ks1i)
/\session(i,nodeV,h,qca,key1i,key2i,ks1i)

end role

goal
secrecy_of sub1
secrecy_of sub2
authentication_on nodeU_nodeV_lt
authentication_on nodeV_nodeU_lt
end goal

environment ()
```