5G P2MP Wireless Backhaul Security

with Formal Verification (Scyther, AVISPA)

Soonchunhyang University

Lab. of Mobile Internet Security (MobiSec)

Ph.D. Course

Jiyoon Kim

74jykim@gmail.com

INDEX

- 5G P2MP Wireless Backhaul Security Protocol
- Formal Verification with Scyther
- Formal Verification with AVISPA
- Comparison: Scyther and AVISPA
- Future Work

5G P2MP Wireless Backhaul

Seoul

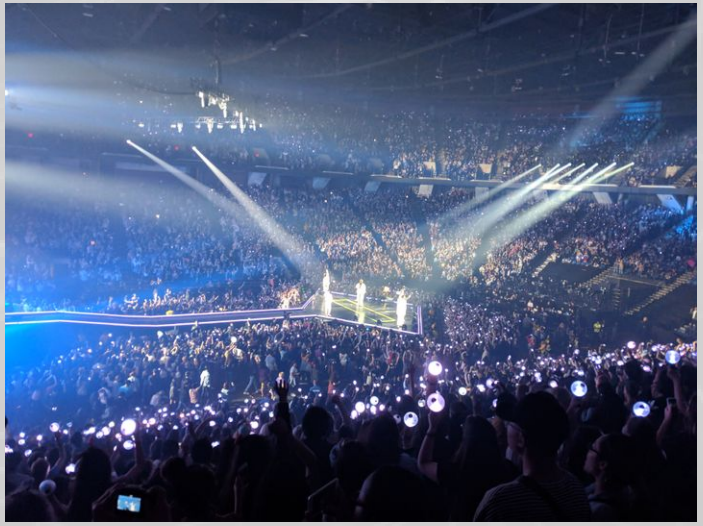
Tokyo

New York

5G P2MP Wireless Backhaul

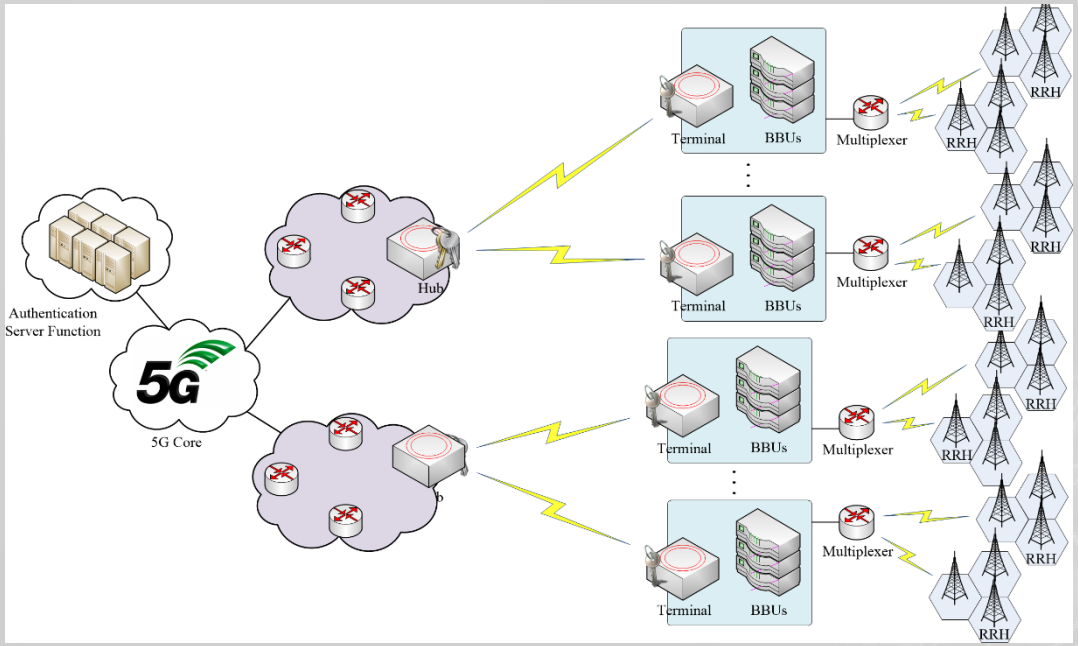
Island

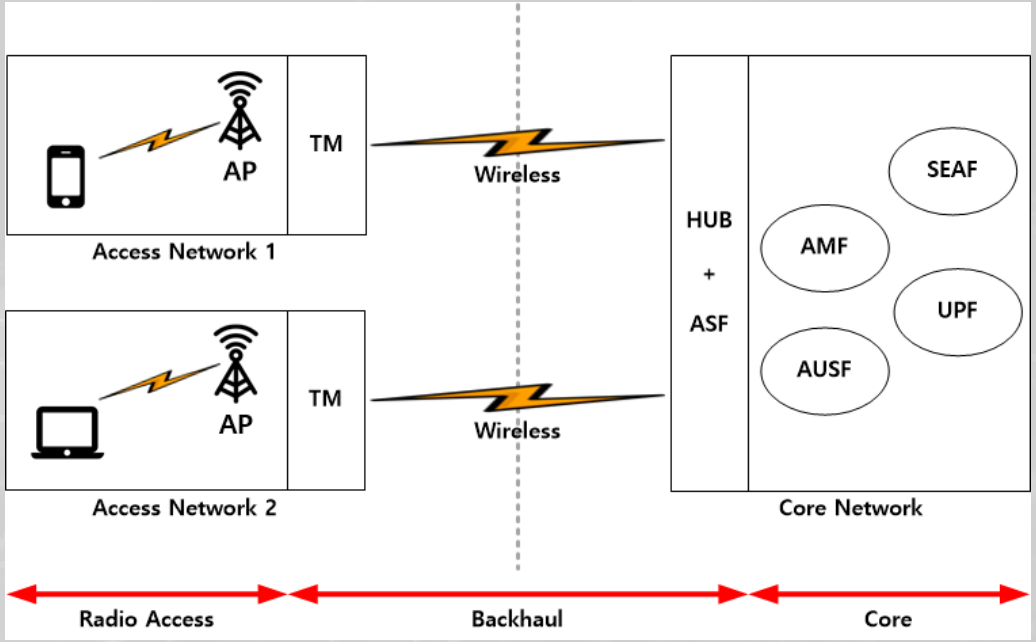
Mountain



Concert

5G P2MP Wireless Backhaul





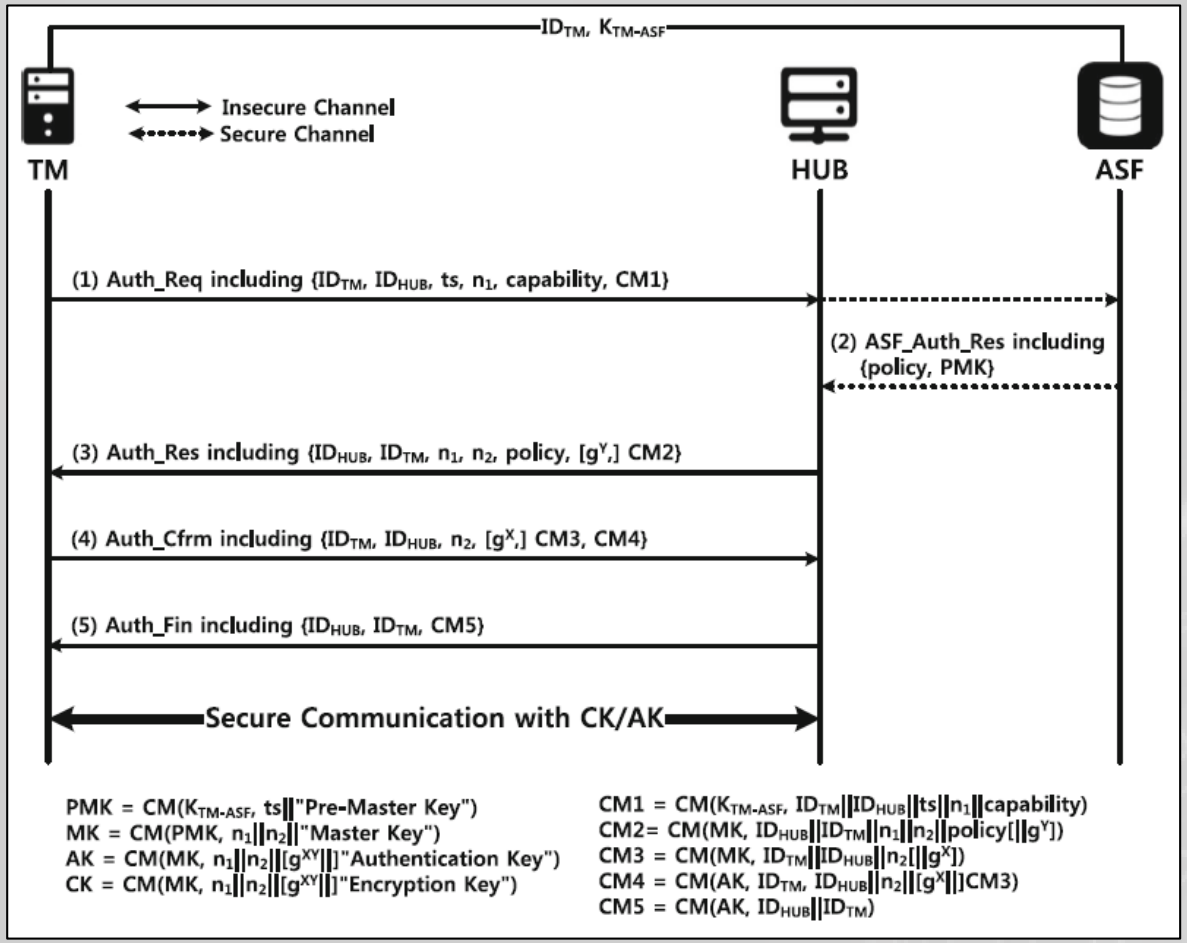
5G P2MP Wireless Backhaul

	Wired Backhaul	Wireless Backhaul
Installation	Difficult	Easy
Maintenance	Difficult	Easy
Price	Cheap	Expensive
Speed	High	Low
Stability	High	Low
Security	High	Low

5G P2MP Wireless Backhaul Security Protocol

순천향대학교
정보보호학과

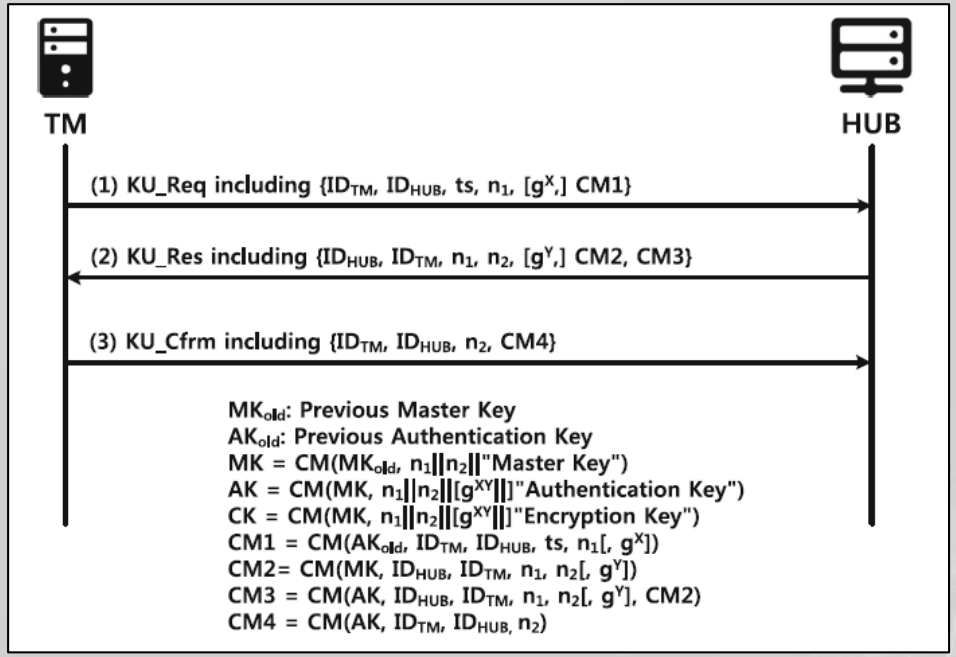
5G P2MP Backhaul Security Protocol



Initial Phase

- TM: Terminal
- HUB: Hub
- ASF: Authentication Server Function
- TM must be pre-registered with ASF.
- In pre-registration, the shared secret key must be shared between two objects.
- We assumed that a secure channel is established between HUB and ASF.

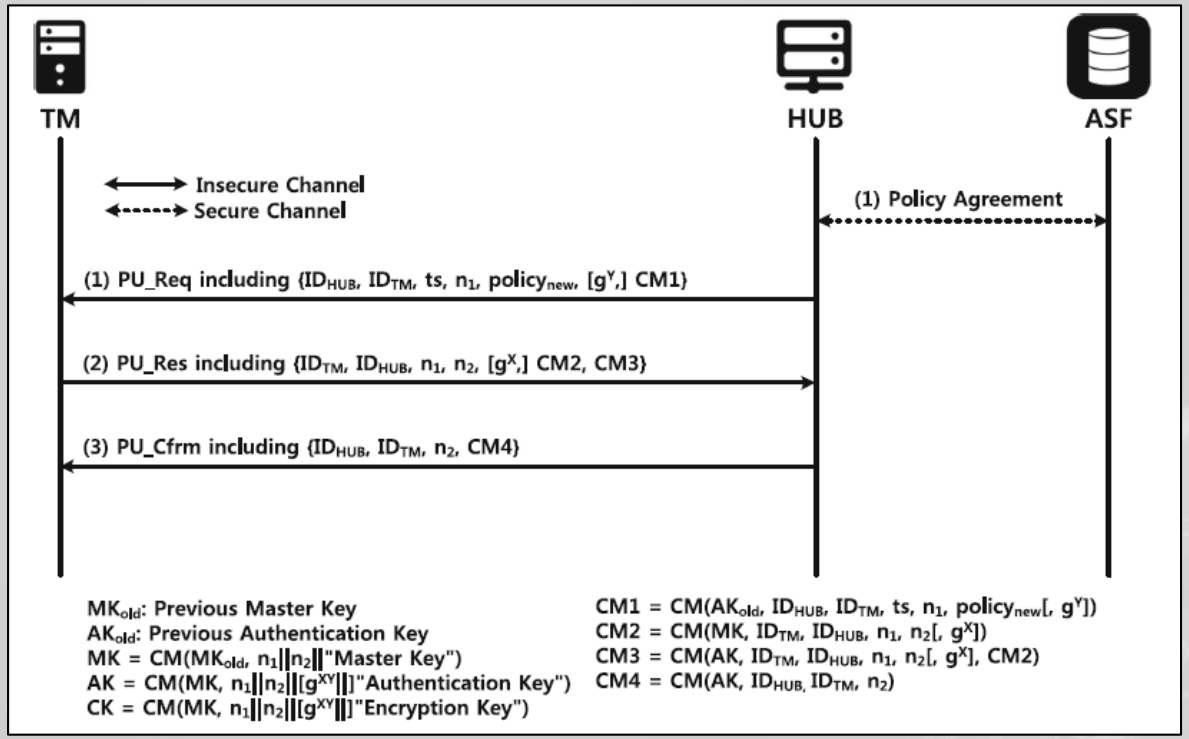
5G P2MP Backhaul Security Protocol



Key Update Phase

- MK: Master session Key
- AK: Authentication Key
- CK: Cipher Key
- Key Update Phase is performed to renew the key MK, AK and CK after initial phase.

5G P2MP Backhaul Security Protocol

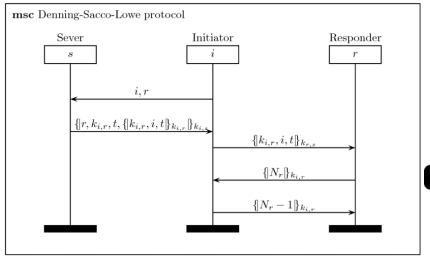


Policy Update Phase

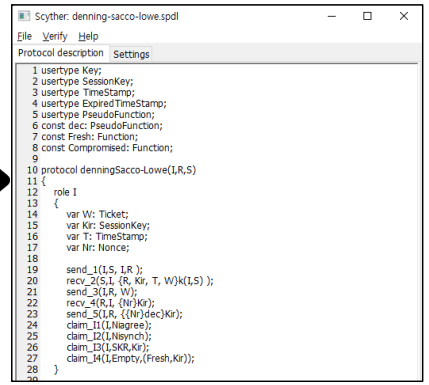
- MK: Master session Key
- AK: Authentication Key
- CK: Cipher Key
- Key Update Phase is performed to renew the key MK, AK and CK after initial phase.

Formal Verification with Scyther

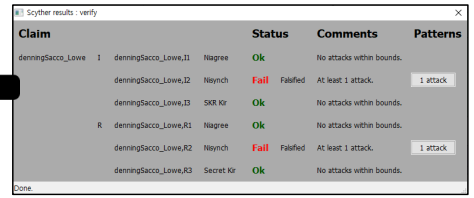
Formal Verification with Scyther



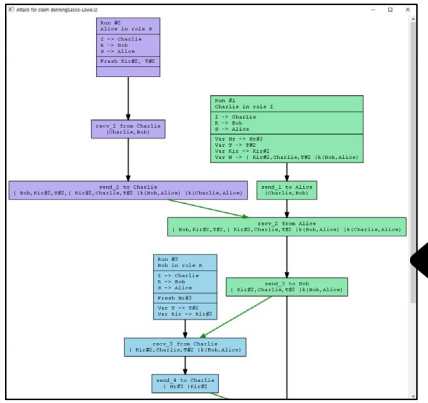
(Step 0) Protocol



(Step 1) Modeling



(Step 2) Verification



(Step 3) Visualization

Formal Verification with Scyther

```
role TM{
        fresh x, ts, n1: Nonce;
        var n2: Nonce;
        var Gy: Ticket;

        send_1(TM, HUB, TM, HUB, ts, n1, cm(Kta, TM, HUB, ts, n1));
        recv_4(HUB, TM, HUB, TM, n1, n2, Gy, cm(MK, HUB, TM, n1, n2, Gy));

        claim(TM, Running, HUB, cm(MK,n1,n2,h(Gy,x)));

        send_!5(TM, HUB, TM, HUB, n2, g(x), cm(cm(MK,n1,n2,h(Gy,x)), TM, HUB, n2, g(x)),
        cm(MK,TM,HUB,n2,g(x),cm(cm(MK,n1,n2,h(Gy,x)),TM,HUB,n2,g(x))));
        recv_!6(HUB, TM, HUB, TM, cm(cm(cm(PMK, n1, n2), n1, n2, h(Gy, x)), HUB, TM));

        claim(TM, Alive);
        claim(TM, Nisynch);
        claim(TM, Niagree);
        claim(TM, Weakagree);
        claim(TM, Commit, HUB, cm(MK,n1,n2,h(Gy,x)));
        claim(TM, Secret, Kta);
        claim(TM, Secret, PMK);
        claim(TM, Secret, MK);
        claim(TM, SKR, cm(MK, n1, n2, h(Gy,x)));
}
```

Role TM(Initial Phase)

```
role HUB{
        fresh y, n2 : Nonce;
        var ts, n1: Nonce;
        var Gx: Ticket;

        recv_1(TM, HUB, TM, HUB, ts, n1, cm(Kta, TM, HUB, ts, n1));
        send_2(HUB, ASF, TM, HUB, ts, n1, cm(Kta, TM, HUB, ts, n1));
        recv_3(ASF, HUB, {PMK}Kha);
        send_4(HUB, TM, HUB, TM, n1, n2, g(y), cm(MK, HUB, TM, n1, n2, g(y)));
        recv_!5(TM, HUB, TM, HUB, n2, Gx, cm(cm(MK,n1,n2,h(Gx,y)), TM, HUB, n2, Gx),
        cm(MK,TM,HUB,n2,Gx,cm(cm(MK,n1,n2,h(Gx,y)),TM,HUB,n2,Gx)));

        claim(HUB, Running, TM, cm(MK,n1,n2,h(Gx,y)));

        send_!6(HUB, TM, HUB, TM, cm(cm(MK, n1, n2, h(Gx, y)), HUB, TM));

        claim(HUB, Alive);
        claim(HUB, Nisynch);
        claim(HUB, Niagree);
        claim(HUB, Weakagree);
        claim(HUB, Commit, TM, cm(MK,n1,n2,h(Gx,y)));
        claim(HUB, Secret, PMK);
        claim(HUB, Secret, cm(PMK, n1, n2));
        claim(HUB, SKR, cm(MK, n1, n2, h(Gx,y)));
}
```

Role HUB(Initial Phase)

```
role ASF{
        var ts, n1: Nonce;

        recv_2(HUB, ASF, TM, HUB, ts, n1, cm(Kta, TM, HUB, ts, n1));
        send_3(ASF, HUB, {PMK}Kha);

        claim(ASF, Secret, Kta);
        claim(ASF, Secret, PMK);
}
```

Role ASF(Initial Phase)

Claim				Status
p2mp	TM	p2mp,TM2	Alive	Ok
		p2mp,TM3	Nisynch	Ok
		p2mp,TM4	Niagree	Ok
		p2mp,TM5	Weakagree	Ok
		p2mp,TM6	Commit HUB,cm(cm(cm(k(TM,ASF),ts),n1,n2),HUB,TM,n1...	Ok
		p2mp,TM7	Secret k(TM,ASF)	Ok
		p2mp,TM8	Secret cm(k(TM,ASF),ts)	Ok
		p2mp,TM9	Secret cm(cm(k(TM,ASF),ts),n1,n2)	Ok
		p2mp,TM10	SKR cm(cm(cm(k(TM,ASF),ts),n1,n2),n1,n2,h(Gy,x))	Ok
	HUB	p2mp,HUB2	Alive	Ok
		p2mp,HUB3	Nisynch	Ok
		p2mp,HUB4	Niagree	Ok
		p2mp,HUB5	Weakagree	Ok
		p2mp,HUB6	Commit TM,cm(cm(cm(k(TM,ASF),ts),n1,n2),HUB,TM,n1,...	Ok
		p2mp,HUB7	Secret cm(k(TM,ASF),ts)	Ok
		p2mp,HUB8	Secret cm(cm(k(TM,ASF),ts),n1,n2)	Ok
		p2mp,HUB9	SKR cm(cm(cm(k(TM,ASF),ts),n1,n2),n1,n2,h(Gx,y))	Ok
	ASF	p2mp,ASF1	Secret k(TM,ASF)	Ok
		p2mp,ASF2	Secret cm(k(TM,ASF),ts)	Ok

Result(Initial Phase)

Formal Verification with Scyther

```
role TM{
        fresh x, ts, n1: Nonce;
        var n2: Nonce;
        var Gy: Ticket;
        secret MKold, AKold;

        send_1(TM,HUB,ts,n1,g(x),cm(AKold,TM,HUB,ts,n1,g(x)));
        recv_!2(HUB,TM,n1,n2,Gy,cm(cm(MK,n1,n2,h(Gy,x)),HUB,TM,n1,n2,Gy),cm(MK,HUB,TM,n1,n2,Gy));

        claim(TM,Running,HUB, cm(MK,n1,n2,h(Gy,x)));

        send_!3(TM,HUB,n2,cm(cm(MK,n1,n2,h(Gy,x)),TM,HUB,n2));

        claim(TM, Alive);
        claim(TM, Nisynch);
        claim(TM, Niagree);
        claim(TM, Weakagree);
        claim(TM, Commit, HUB, cm(MK,n1,n2,h(Gy,x)));
        claim(TM, Secret, MKold);
        claim(TM, Secret, AKold);
        claim(TM, SKR, MK);
        claim(TM, SKR, cm(MK,n1,n2,h(Gy,x)));
}
```

Role TM(Key Update Phase)

```
role HUB{
        fresh y, n2: Nonce;
        var ts, n1: Nonce;
        var Gx: Ticket;
        secret MKold, AKold;

        recv_1(TM,HUB,ts,n1,Gx,cm(AKold,TM,HUB,ts,n1,Gx));

        claim(HUB, Running, TM, cm(MK,n1,n2,h(Gx,y)));

        send_!2(HUB,TM,n1,n2,g(y),cm(cm(MK,n1,n2,h(Gx,y)),HUB,TM,n1,n2,g(y)),cm(MK,HUB,TM,n1,n2,g(y)));
        recv_!3(TM,HUB,n2,cm(cm(MK,n1,n2,h(Gx,y)),TM,HUB,n2));

        claim(HUB, Alive);
        claim(HUB, Nisynch);
        claim(HUB, Niagree);
        claim(HUB, Weakagree);
        claim(HUB, Commit, TM, cm(MK,n1,n2,h(Gx,y)));
        claim(HUB, Secret, MKold);
        claim(HUB, Secret, AKold);
        claim(HUB, SKR, MK);
        claim(HUB, SKR, cm(MK,n1,n2,h(Gx,y)));
}
```

Role HUB(Key Update Phase)

Claim				Status
p2mpkeyupdate	TM	p2mpkeyupdate,TM2	Alive	Ok
		p2mpkeyupdate,TM3	Nisynch	Ok
		p2mpkeyupdate,TM4	Niagree	Ok
		p2mpkeyupdate,TM5	Weakagree	Ok
		p2mpkeyupdate,TM6	Commit HUB,cm(cm(MKold,n1,n2),n1,n2,h(Gy,x))	Ok
		p2mpkeyupdate,TM7	Secret MKold	Ok
		p2mpkeyupdate,TM8	Secret AKold	Ok
		p2mpkeyupdate,TM9	SKR cm(MKold,n1,n2)	Ok
		p2mpkeyupdate,TM10	SKR cm(cm(MKold,n1,n2),n1,n2,h(Gy,x))	Ok
	HUB	p2mpkeyupdate,HUB2	Alive	Ok
		p2mpkeyupdate,HUB3	Nisynch	Ok
		p2mpkeyupdate,HUB4	Niagree	Ok
		p2mpkeyupdate,HUB5	Weakagree	Ok
		p2mpkeyupdate,HUB6	Commit TM,cm(cm(MKold,n1,n2),n1,n2,h(Gx,y))	Ok
		p2mpkeyupdate,HUB7	Secret MKold	Ok
		p2mpkeyupdate,HUB8	Secret AKold	Ok
		p2mpkeyupdate,HUB9	SKR cm(MKold,n1,n2)	Ok
		p2mpkeyupdate,HUB10	SKR cm(cm(MKold,n1,n2),n1,n2,h(Gx,y))	Ok

Result(Key Update Phase)

Formal Verification with Scyther

```
role TM{
        fresh x, n2: Nonce;
        var ts, n1: Nonce;
        var Gy: Ticket;
        secret AKold, MKold, policy;

        recv_1(HUB,TM,ts,n1,policy,Gy,cm(AKold,HUB,TM,ts,n1,policy,Gy));

        claim(TM,Running,HUB, cm(MK,n1,n2,h(Gy,x)));

        send_!2(TM,HUB,n1,n2,g(x),cm(cm(MK,n1,n2,h(Gy,x)),TM,HUB,n1,n2,g(x)),
        cm(MK,TM,HUB,n1,n2,g(x),cm(cm(MK,n1,n2,h(Gy,x)),TM,HUB,n1,n2,g(x))));
        recv_!3(HUB,TM,n2,cm(cm(MK,n1,n2,h(Gy,x)),HUB,TM,n2));

        claim(TM, Alive);
        claim(TM, Nisynch);
        claim(TM, Niagree);
        claim(TM, Weakagree);
        claim(TM, Commit, HUB, cm(MK,n1,n2,h(Gy,x)));
        claim(TM, Secret, MKold);
        claim(TM, Secret, AKold);
        claim(TM, SKR, MK);
        claim(TM, SKR, cm(MK,n1,n2,h(Gy,x)));
}
```

Role TM(Policy Update Phase)

```
role HUB{
        fresh ts, y, n1: Nonce;
        var n2: Nonce;
        var Gx: Ticket;
        secret AKold, MKold, policy;

        send_1(HUB,TM,ts,n1,policy,g(y),cm(AKold,HUB,TM,ts,n1,policy,g(y)));
        recv_!2(TM,HUB,n1,n2,Gx,cm(cm(MK,n1,n2,h(Gx,y)),TM,HUB,n1,n2,Gx),
        cm(MK,TM,HUB,n1,n2,Gx,cm(cm(MK,n1,n2,h(Gx,y)),TM,HUB,n1,n2,Gx)));

        claim(HUB, Running, TM, cm(MK,n1,n2,h(Gx,y)));

        send_!3(HUB,TM,n2,cm(cm(MK,n1,n2,h(Gx,y)),HUB,TM,n2));

        claim(HUB, Alive);
        claim(HUB, Nisynch);
        claim(HUB, Niagree);
        claim(HUB, Weakagree);
        claim(HUB, Commit, TM, cm(MK,n1,n2,h(Gx,y)));
        claim(HUB, Secret, MKold);
        claim(HUB, Secret, AKold);
        claim(HUB, SKR, MK);
        claim(HUB, SKR, cm(MK,n1,n2,h(Gx,y)));
}
```

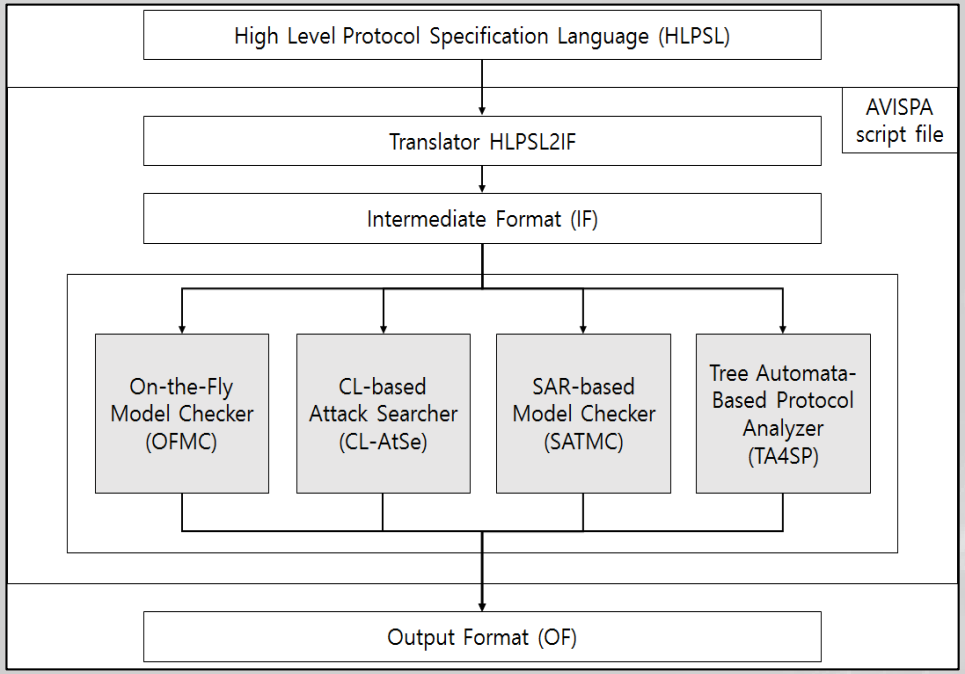
Role HUB(Policy Update Phase)

Claim				Status
p2mppolicyupdate	TM	p2mppolicyupdate,TM2	Alive	Ok
		p2mppolicyupdate,TM3	Nisynch	Ok
		p2mppolicyupdate,TM4	Niagree	Ok
		p2mppolicyupdate,TM5	Weakagree	Ok
		p2mppolicyupdate,TM6	Commit HUB,cm(cm(MKold,n1,n2),n1,n2,h(Gy,x))	Ok
		p2mppolicyupdate,TM7	Secret MKold	Ok
		p2mppolicyupdate,TM8	Secret AKold	Ok
		p2mppolicyupdate,TM9	SKR cm(MKold,n1,n2)	Ok
		p2mppolicyupdate,TM10	SKR cm(cm(MKold,n1,n2),n1,n2,h(Gy,x))	Ok
	HUB	p2mppolicyupdate,HUB2	Alive	Ok
		p2mppolicyupdate,HUB3	Nisynch	Ok
		p2mppolicyupdate,HUB4	Niagree	Ok
		p2mppolicyupdate,HUB5	Weakagree	Ok
		p2mppolicyupdate,HUB6	Commit TM,cm(cm(MKold,n1,n2),n1,n2,h(Gx,y))	Ok
		p2mppolicyupdate,HUB7	Secret MKold	Ok
		p2mppolicyupdate,HUB8	Secret AKold	Ok
		p2mppolicyupdate,HUB9	SKR cm(MKold,n1,n2)	Ok
		p2mppolicyupdate,HUB10	SKR cm(cm(MKold,n1,n2),n1,n2,h(Gx,y))	Ok

Result(Policy Update Phase)

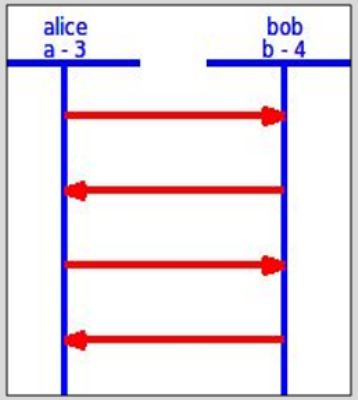
Formal Verification with AVISPA

Formal Verification with AVISPA



(Step 1) Modeling

```
role environment()
def=

  const na_nb1, na_nb2 : protocol_id,
      h, prf, keygen : function,
      a, b          : agent,
      ka, kb, ki, ks : public_key

  intruder_knowledge = { a, b, ka, kb, ks, ki, inv(ki),
                         {i.ki}_(inv(ks)) }

  composition
          session(a,b,ka,kb,ks,h,prf,keygen)
  ∧       session(a,i,ka,ki,ks,h,prf,keygen)
  ∧       session(i,b,ki,kb,ks,h,prf,keygen)

end role
```



(Step 2) Verification

```
SUMMARY
 SAFE

DETAILS
 BOUNDED_NUMBER_OF_SESSIONS
 TYPED_MODEL

PROTOCOL
 /home/span/span/testsuite/results/TLS.if

GOAL
 As Specified

BACKEND
 CL-AtSe

STATISTICS

 Analysed    : 253 states
 Reachable   : 113 states
 Translation: 0.01 seconds
 Computation: 0.01 seconds
```

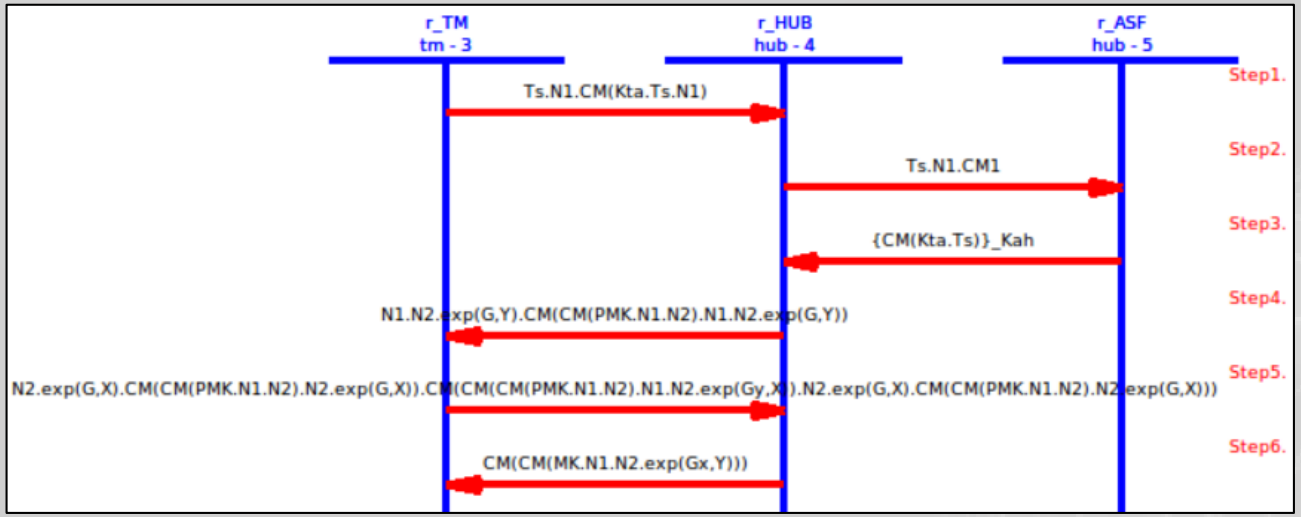
*ATSE

```
% OFMC
% Version of 2006/02/13
SUMMARY
 SAFE
DETAILS
 BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
 /home/span/span/testsuite/results/TLS.if
GOAL
 as_specified
BACKEND
 OFMC
COMMENTS
STATISTICS
 parseTime: 0.00s
 searchTime: 0.12s
 visitedNodes: 106 nodes
 depth: 8 plies
```

*OFMC

Formal Verification with AVISPA

- Initial Phase



Protocol Simulation

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/abc.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.02s
  visitedNodes: 22 nodes
  depth: 7 plies
```

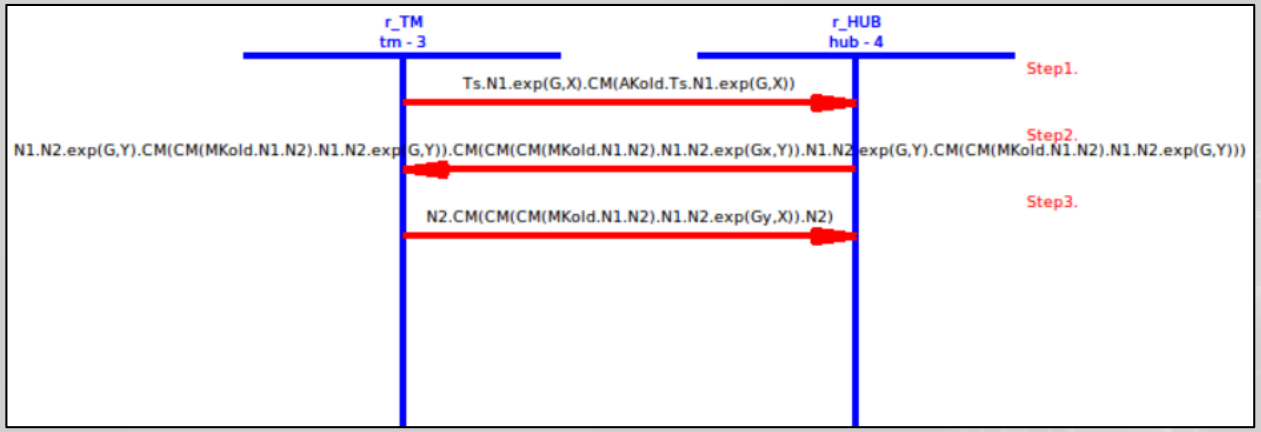
OFMC Result

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/span/span/testsuite/results/abc.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed    : 8 states
  Reachable   : 4 states
  Translation: 0.02 seconds
  Computation: 0.00 seconds
```

CL-AtSe Result

Formal Verification with AVISPA

- Key Update Phase



Protocol Simulation

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/P2MP_KU.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.00s
  visitedNodes: 4 nodes
  depth: 2 plies
```

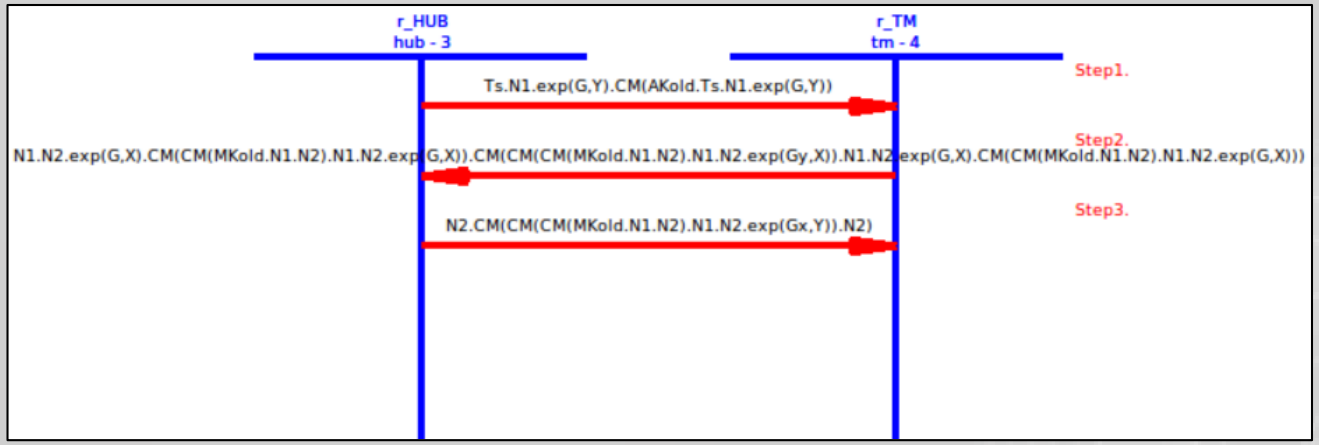
OFMC Result

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/span/span/testsuite/results/P2MP_KU.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed    : 4 states
  Reachable   : 1 states
  Translation: 0.00 seconds
  Computation: 0.00 seconds
```

CL-AtSe Result

Formal Verification with AVISPA

- Policy Update Phase



Protocol Simulation

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/P2MP_PU.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.00s
  visitedNodes: 4 nodes
  depth: 2 plies
```

OFMC Result

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/span/span/testsuite/results/P2MP_PU.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed    : 4 states
  Reachable   : 1 states
  Translation: 0.00 seconds
  Computation: 0.00 seconds
```

CL-AtSe Result

Comparison: Scyther and AVISPA

Comparison: Scyther and AVISPA

	AVISPA	Scyther
Environment	Ubuntu	Linux / Windows / Mac OS
Last Version	Version 1.6 (September 2017)	V 1.1.3 (April 2014)
Type	Model Checker	Model Checker
Module	4 (OFMC, CL-AtSe, SATMC, TA4SP)	1
Simulation	Yes	No
Freshness	No	Yes
Unbounded Session	Partially Support	Support
Command	Simple	Various
Difficulty	High	Low

Comparison: Scyther and AVISPA

```
role r_HUB(
        TM,HUB,AS                       : agent,
        G                       : text,
        CM                      : hash_func,
        Kah                     : symmetric_key,
        SND, RCV, SND2, RCV2            : channel(dy))
played_by HUB def=

local
        State : nat,
        Kta : symmetric_key,
        Ts, N1, N2, Y : text,
        Gx, Gy, Gxy : message,
        PMK : hash(symmetric_key.text),
        MK : hash(hash(symmetric_key.text).text.text),
        AK : hash(hash(hash(symmetric_key.text).text.text).text.text.message),
        CM1 : hash(symmetric_key.text.text),
        CM2 : hash(hash(hash(symmetric_key.text).text.text).text.text.message),
        CM3 : hash(hash(hash(symmetric_key.text).text.text).text.message),
        CM4 : hash(hash(hash(hash(symmetric_key.text).text.text).text.text.message).
        text.message.hash(hash(hash(symmetric_key.text).text.text).text.message)),
        CM5 : hash(hash(hash(hash(symmetric_key.text).text.text).text.text.message))

init
        State := 1

transition
1.      State = 1               /\/ RCV(Ts'.N1'.CM1') =|>
        State' := 3                     /\/ SND2(Ts'.N1'.CM1')

3.      State = 3               /\/ RCV2({PMK'}_Kah) =|>
        State' := 5                     /\/ N2' := new()
                                /\/ Y' := new()
                                /\/ Gy' := exp(G,Y')
                                /\/ MK' := CM(PMK'.N1.N2')
                                /\/ CM2' := CM(MK'.N1.N2'.Gy')
                                /\/ SND(N1.N2'.Gy'.CM2')
                                /\/ secret(MK',sec2,{HUB,TM})
                                /\/ witness(HUB,TM,auth1,N1)
```

AVISPA

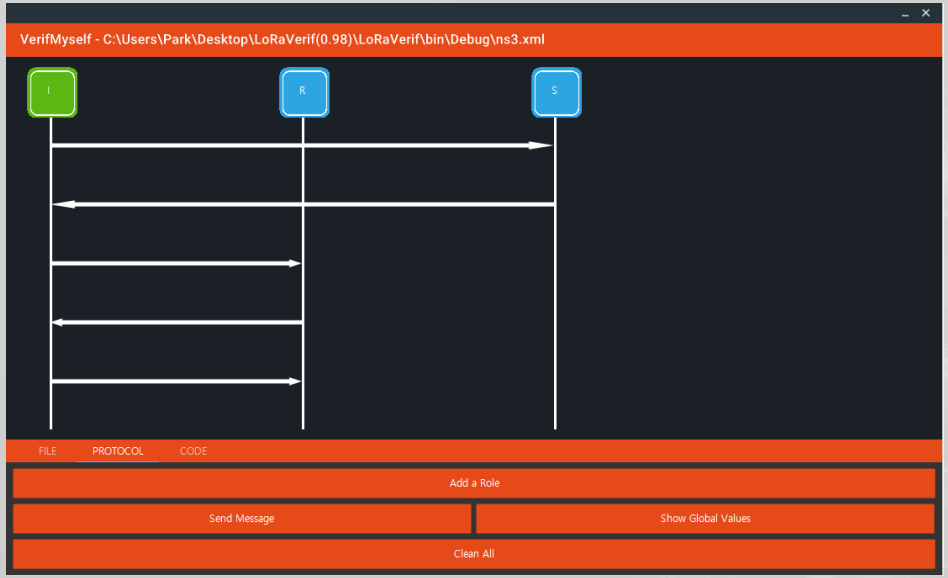
```
role HUB{
        fresh y, n2 : Nonce;
        var ts, n1: Nonce;
        var Gx: Ticket;

        recv_1(TM, HUB, TM, HUB, ts, n1, cm(Kta, TM, HUB, ts, n1));
        send_2(HUB, ASF, TM, HUB, ts, n1, cm(Kta, TM, HUB, ts, n1));
        recv_3(ASF, HUB, {PMK}Kha);
        send_4(HUB, TM, HUB, TM, n1, n2, g(y), cm(MK, HUB, TM, n1, n2, g(y)));
        recv_!5(TM, HUB, TM, HUB, n2, Gx, cm(cm(MK,n1,n2,h(Gx,y)), TM, HUB, n2, Gx),
        cm(MK,TM,HUB,n2,Gx,cm(cm(MK,n1,n2,h(Gx,y)),TM,HUB,n2,Gx)));

        claim(HUB, Running, TM, cm(MK,n1,n2,h(Gx,y)));

        send_!6(HUB, TM, HUB, TM, cm(cm(MK, n1, n2, h(Gx, y)), HUB, TM));

        claim(HUB, Alive);
        claim(HUB, Nisynch);
        claim(HUB, Niagree);
        claim(HUB, Weakagree);
        claim(HUB, Commit, TM, cm(MK,n1,n2,h(Gx,y)));
        claim(HUB, Secret, PMK);
        claim(HUB, Secret, cm(PMK, n1, n2));
        claim(HUB, SKR, cm(MK, n1, n2, h(Gx,y)));
}
```

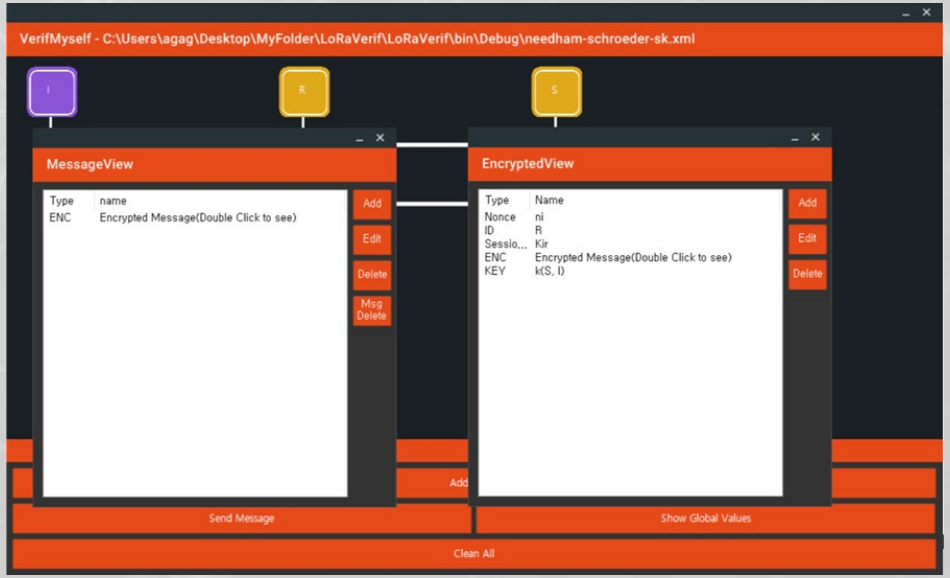
Scyther

Future Work

Future Work

- Development of Automation Scripting Tool
  - Easy Expression
  - Visualization (Protocol, Attack model)
  - Conversion (AVISPA, Scyther, TAMARIN)





Thank you