fixed version

Protocol Purpose

Sender invariance (authentication assuming that the first message is not tampered with)

Definition Reference

http://www.ietf.org/internet-drafts/draft-bradner-pbk-frame-06.txt

Model Authors

- Daniel Plasto for Siemens CT IC 3, 2004
- Sebastian Mödersheim, ETH Zürich

Alice&Bob style

```
A -> B: A, PK_A, hash(PK_A)
A -> B: {***tag1***,Msg}inv(PK_A), hash(PK_A)
B -> A: Nonce
A -> B: {***tag2***,Nonce}inv(PK_A)
```

Problems considered: 1

Attacks Found

Initially, we demanded (strong) authentication, but this does of course not hold as there is nothing that guarantees freshness, until the agent generates a new public key, as in the following replay attack, which is possible after observing a session between honest agents *a* and *b* using *Msg*(1) as the exchanged message.

```
i -> (a,3): start
(a,3) -> i: b,{tag1,Msg(1)}inv(pk_a),f(pk_a)
i -> (b,3): b,{tag1,Msg(1)}inv(pk_a),f(pk_a)
(b,3) -> i: Nonce(3)
i -> (a,3): Nonce(3)
```

```
     (a,3) -> i: {tag2,Nonce(3)}inv(pk_a)
     i -> (b,3): {tag2,Nonce(3)}inv(pk_a)

     i -> (a,6): start
     (a,6) -> i: b,{tag1,Msg(4)}inv(pk_a),f(pk_a)
     i -> (b,6): b,{tag1,Msg(1)}inv(pk_a),f(pk_a)
     (b,6) -> i: Nonce(6)
     i -> (a,6): Nonce(6)
     (a,6) -> i: {tag2,Nonce(6)}inv(pk_a)
     i -> (b,6): {tag2,Nonce(6)}inv(pk_a)
```

Further Notes

Prevents the attack of the initial version by tagging the nonce before signing it. This version was only provide to demonstrate that the protocol cannot ensure strong authentication.

HLPSL Specification

```
role alice (A,B        : agent,
            SND,RCV    : channel(dy),
            Hash       : function,
            PK_A       : public_key,
            Tag1,Tag2  : text)
played_by A
def=

  local
    State      : nat,
    Msg        : text,
    Nonce      : text

  init  State := 0

  transition
```

```
 1. State  = 0 /\ RCV(start) =|>
    State':= 2 /\ Msg' := new()
               /\ SND(B.{Tag1.Msg'}_inv(PK_A).Hash(PK_A))
               /\ witness(A,A,msg,Msg')

 3. State  = 2 /\ RCV(Nonce') =|>
    State':= 4 /\ SND({Tag2.Nonce'}_inv(PK_A))

end role
```

```
role bob (B,A         : agent,
          SND,RCV     : channel(dy),
          Hash        : function,
          PK_A        : public_key,
          Tag1,Tag2   : text)
played_by B
def=

  local
    State      : nat,
    Nonce      : text,
    Msg        : text

  init State := 1

  transition

 1. State  = 1 /\ RCV(B.{Tag1.Msg'}_inv(PK_A).Hash(PK_A)) =|>
    State':= 5 /\ Nonce' := new()
               /\ SND(Nonce')

 3. State  = 5 /\ RCV({Tag2.Nonce}_inv(PK_A)) =|>
    State':= 7 /\ request(A,A,msg,Msg)

end role
```

```
role session(A,B        : agent,
             Hash       : function,
             PK_A       : public_key,
             Tag1,Tag2  : text)
def=

  local SNDA,RCVA,SNDB,RCVB  : channel (dy)

  composition

     alice(A,B,SNDA,RCVA,Hash,PK_A,Tag1,Tag2)
  /\ bob(B,A,SNDB,RCVB,Hash,PK_A,Tag1,Tag2)

end role
```

```
role environment() def=

  const
    a,b            : agent,
    f              : function,
    msg            : protocol_id,
    pk_a,pk_b,pk_i : public_key,
    tag1,tag2      : text

  intruder_knowledge = {a,b,f,pk_a,pk_b,pk_i,inv(pk_i)}

  composition
    session(a,b,f,pk_a,tag1,tag2)
 /\ session(a,b,f,pk_a,tag1,tag2)

end role
```

```
goal

  %Alice authenticates Alice on msg
  authentication_on msg

end goal
```

```
environment()
```

References