SPIIRAS

Verification of Entrusting Protocol using AVISPA and Isabelle

Sergey Reznick

Computer Security Research Group,
St. Petersburg Institute for Informatics and Automation of Russian Academy of Sciences

Agenda

- *Responsibility areas*
- Example of the confidentiality verification using AVISPA
- Example of the authentication verification using Isabelle

RE-TRUST Workshop, June 19, 2008

Responsibility areas (1/2)

- General principle: use AVISPA wherever possible, use Isabelle otherwise
- Confidentiality (man-in-the-middle attack): AVISPA
- Authenticity (man-in-the-middle attack): AVISPA

Responsibility areas (2/2)

- Confidentiality, authenticity (man-in-the-end attack): AVISPA, Isabelle
- Attacks not expressed in terms of the confidentiality and/or authenticity: Isabelle
  - Time-related attacks
  - Attacks involving modules/tags properties

State-of-the-art

- Confidentiality (man-in-the-middle case) is modelled using AVISPA: will be demonstrated
- Isabelle model for compound cases is under development
- Isabelle development for simpler cases: evaluation purpose

Agenda

- *Responsibility areas*
- *Example of the confidentiality verification using AVISPA*
- Example of the authentication verification using Isabelle

Model assumptions

- Model for all details would be too huge
- Partial model: based on assumptions
- Assumptions about server:
  - Trusted server
  - Untrusted server

AVISPA model: server's step 1

```
role server (S, C: agent, K : symmetric_key,
SND, RCV: channel (dy))
.............
transition
    0.  State  = 0 /\ RCV(start) =|>
              State':= 2 /\  SND({Module0}_K)
.....................................
end role
```

AVISPA model: client's step 1

```
role client(S, C: agent, K : symmetric_key, SND,
RCV: channel (dy))
.....................
transition
   1.  State  = 1  /\ RCV({Module0}_K) =|>
              State':=3 /\ SND({Tag0}_K) /\
secret(Tag0, p, {S,C})
end role
```


secret() predicate

- Secret (Tag0, p, {S,C})
  - Tag0 – secret data
  - p – binding to the goals set
  - {S,C} – set of entities allowed to know the secret

environment

- role environment() def=
  .....
  intruder_knowledge = {s,c,ksi,kic}
  .....
  composition
  session(s,c,ksc)
  /\session(s,i,ksi)
  /\session(i,c,kic)
  end role

Goal

goal
secrecy_of p
end goal

Verification using CL-AtSE (1/2)

- SUMMARY
  UNSAFE
  .....

  ATTACK TRACE
  i -> (s,6): start
  (s,6) -> i: {dummy_msg}_ksi

  i -> (s,3): start
  (s,3) -> i: {dummy_msg}_ksc

Verification using CL-AtSE

- i -> (c,10): {dummy_msg}_kic
  (c,10) -> i: {dummy_nonce}_kic
  &
  Secret(dummy_nonce,set_65); Add i to set_65; Add c to set_65;

  i -> (c,4): {dummy_msg}_ksc
  (c,4) -> i: {dummy_nonce}_ksc
  & Secret(dummy_nonce,set_59); Add s to set_59; Add c to set_59;

Trace evaluation (1/3)

Intruder connects with server as client:

----

i -> (s,6): start

(s,6) -> i: {dummy_msg}_ksi

Trace evaluation (2/3)

Intruder convinces client to connect to the server:

----

i -> (s,3): start

(s,3) -> i: {dummy_msg}_ksc

Trace evaluation (3/3)

Intruder replaces server for honest client, send a module and receives a tag:

---

i -> (c,10): {dummy_msg}_kic

(c,10) -> i: {dummy_nonce}_kic

& Secret(dummy_nonce,set_65); Add i to set_65; Add c to set_65;

Critical assumption

- Important assumption is untrusted server
- Corresponding session is «session(i,c,kic)»
- After removal this session result is SAFE

SPAN

- Let's model protocol behaviour using SPAN

Agenda

- Responsibility areas
  - Example of the confidentiality verification using AVISPA
  - *Example of the authentication verification using Isabelle*

Isabelle output

- Text of successfully proved statements
- Isabelle reports about proof states
- Messages about proof fails (if available)

Protocol representation (1/3)

```
inductive_set retrust :: "event list set"
  where
        (*Initial trace is empty*)
   Nil:  "[] \<in> retrust"

        (*Alice initiates a protocol run*)
 | RT1:  "evs1 \<in> retrust ==> Says A B (Agent
   A) # evs1 \<in> retrust"
```

Protocol representation (2/3)

(*Bob responds to Alice's message with a module.*)

| RT2: "[| evs2 \<in> retrust; Says A B (Agent A) \<in> set evs2 |]

==> Says B A (Crypt (shrK B) (Nonce NB)) # evs2 \<in> retrust"

Protocol representation (3/3)

(*Bob responds to Alice's message with a module.*)

| RT3: "[| evs3 \<in> retrust;
Says A B (Agent A) \<in> set evs3;
Says B' A (Crypt (shrK B) (Nonce NB)) \<in> set evs3 |]
==> Says A B (Crypt (shrK A) (Nonce NA)) # evs3 \<in> retrust"

Authentication theorems (1/2)

- lemma NB_Crypt_imp_Alice_msg2:
- "[| Crypt (shrK B) (Nonce NB) \<in> parts (spies evs);
- B \<notin> bad; evs \<in> retrust |]
- ==> \<exists>A. Says B A (Crypt (shrK B) (Nonce NB)) \<in> set evs"
- by (erule rev_mp, erule retrust.induct, force, simp_all, blast+)

Authentication theorems (2/2)

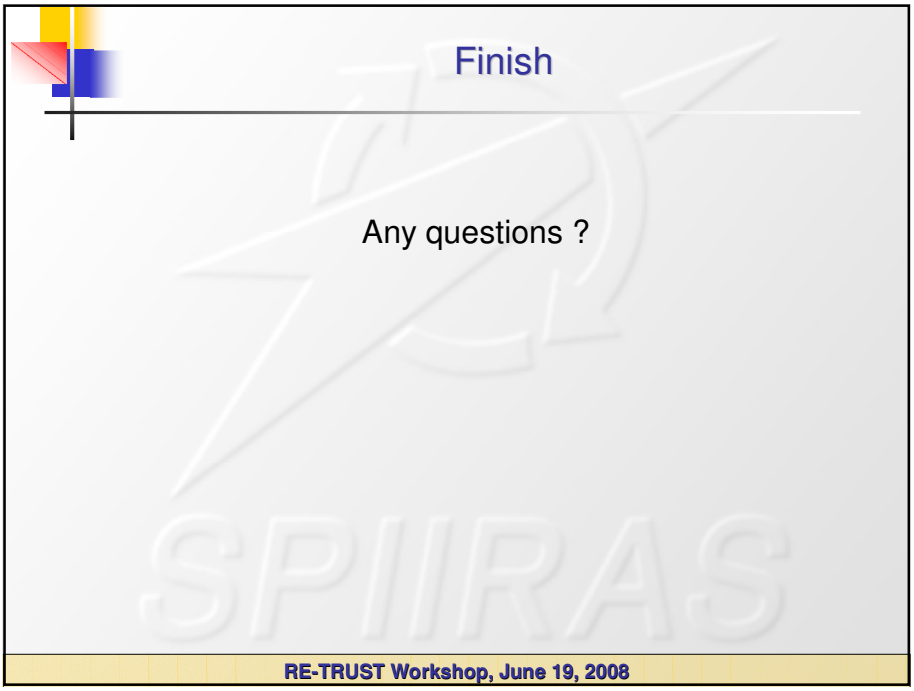
lemma NB_Crypt_imp_Alice_msg3:
"[| Crypt (shrK A) (Nonce NA) \<in> parts (spies evs);
A \<notin> bad; evs \<in> retrust |]
==> \<exists>B. Says A B (Crypt (shrK A) (Nonce NA)) \<in> set evs"
by (erule rev_mp, erule retrust.induct, force, simp_all, blast+)

Authentication proof output

```
lemma
  NB_Crypt_imp_Alice_msg2:
   [| Crypt (shrK ?B) (Nonce ?NB) : parts (knows
  Spy ?evs); ?B ~: bad;
      ?evs : retrust |]
   ==> EX A. Says ?B A (Crypt (shrK ?B) (Nonce
  ?NB)) : set ?evs
Search depth = 0
Search depth = 1
Search depth = 0
Search depth = 1
```


Plans for future

- Prove more complicated properties balancing Isabelle and AVISPA capabilities
- Develop general methodology of verification tools interaction to cover different aspects of the protocol verification
- Consider use of PRISM to evaluate probabilistic properties


Finish
Any questions ?
SPIIRAS
RE-TRUST Workshop, June 19, 2008