Special topics in Logic and Security I

Master Year II, Sem. I, 2022-2023

Ioana Leuștean
FMI, UB

Formal analysis of security protocols

- In formal analysis we define and analyze a protocol within a consistent mathematical theory.
- One studies abstract versions of real protocols (for example the (real, computer-network authentication) protocol Kerberos is based on the (academic) protocol Needham-Scroeder.
- Verification based on:
  - epistemic logics, for example BAN logic, 1990
  - model-checking tools: Proverif, AVISPA, Scyther, Tamarin, ...
  - ...

In the following we shall present the *operational semantics approach* from [OSVSP]:

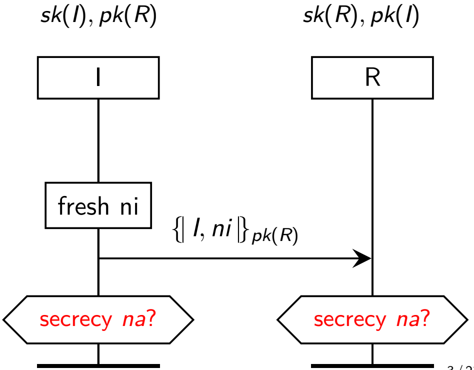
- Cremers C. and Mauw S. Operational Semantics and Verification of Security Protocols. Springer, 2012.
- Cremers, C. J. F. (2006). Scyther : semantics and verification of security protocols Eindhoven: Technische Universiteit Eindhoven DOI: 10.6100/IR614943

Example: OSS Protocol

One-Sided Secrecy Protocol (OSS)

$I \longrightarrow R : \{\!|\, I, ni \,|\!\}_{pk(R)}$

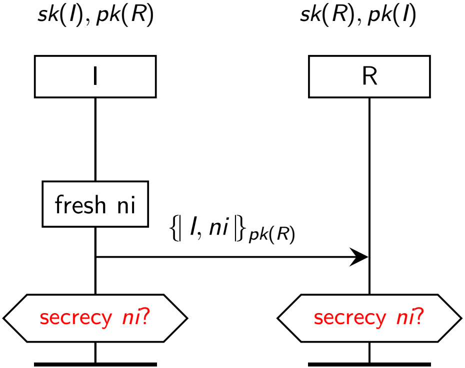
What does secrecy mean?



Example: OSS Protocol

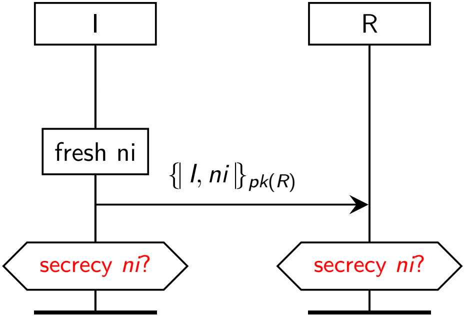
What does secrecy ni mean?

[OSVSP]
We say that the claim secrecy ni holds *for the role i* whenever a run of the *i* role is completed with trusted communication partners and the nonce *ni* generated in the run does not become known to the intruder



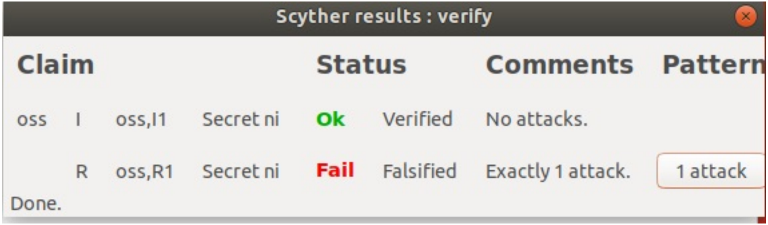
Scyther: OSS Protocol

```
protocol oss(I,R)
{role I
{fresh ni: Nonce;
send_1(I,R, {I,ni}pk(R) );
claim(I,Secret,ni);}

role R
{var ni: Nonce;
recv_1(I,R, {I,ni}pk(R) );
claim(R,Secret,ni);
}}
```

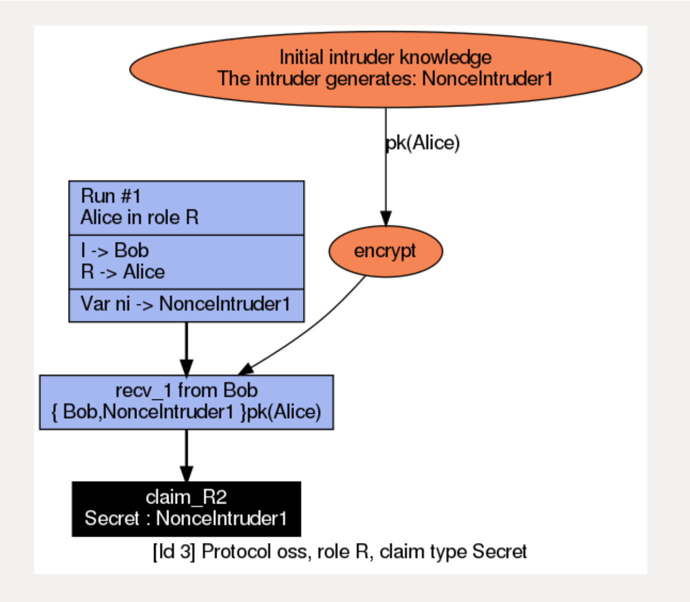


Scyther: OSS Protocol

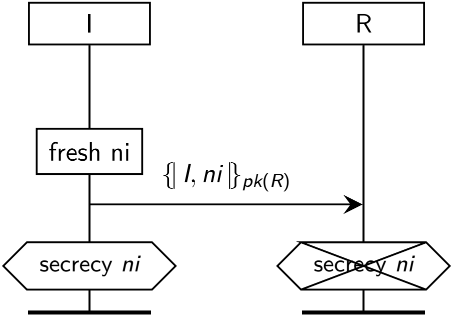
```
protocol oss(I,R)
{role I
{fresh ni: Nonce;
send_1(I,R, {I,ni}pk(R) );
claim(I,Secret,ni);}

role R
{var ni: Nonce;
recv_1(I,R, {I,ni}pk(R));
claim(R,Secret,ni);
}}
```



Scyther: OSS Protocol



Scyther: OSS Protocol



The claims are locally analyzed!

Formal analysis of security protocols

- A formal approach should:
  - specify the security protocol,
  - provide a model for agents,
  - provide a model for communication,
  - provide a threat model,
  - express the security requirements.

- In the following, we shall use a *many-sorted language:*
  - *the roles and the messages are represented by terms,*
  - *a protocol is defined by specifying each role,*
  - *the (adversary) knowledge is derived using a deduction system,*
  - *the protocol execution is defined as a (complex) transition system,*
  - *the security properties are formally defined and analyzed.*

Basic elements

- There are few types of variables and constants:
  - *Var* : $V, X, Y, Z, \ldots$ (variables for messages),
  - *Fresh* : $ni, nr, sessionkey, \ldots$ (local constants, freshly generated for each instance of a role),
  - *Role* : $i, r, s, \ldots$ (variables for roles: initiator, respondent, server etc.)

- *Func* is a set of function symbols
  - each symbol has a fixed arity;
  - the global constants are functions of arity zero;
  - example: hash functions.

Terms: *RoleTerm*

We use *role terms to specify: messages, nonces, roles, keys.*

$$
\begin{aligned}
\mathit{RoleTerm} \quad ::= \quad & \mathit{Var} \mid \mathit{Fresh} \mid \mathit{Role} \\
& \mid \mathit{Func}\,(\mathit{RoleTerm}^*) \\
& \mid (\mathit{RoleTerm}, \mathit{RoleTerm}) \\
& \mid \{\!|\ \mathit{RoleTerm}\ |\!\}_{\mathit{RoleTerm}} \\
& \mid \mathit{sk}(\mathit{RoleTerm}) \mid \mathit{pk}(\mathit{RoleTerm}) \mid k(\mathit{RoleTerm}, \mathit{RoleTerm})
\end{aligned}
$$

We denote $\{\!|\ (t_1, t_2)\ |\!\}$ *by* $\{\!|\ t_1, t_2\ |\!\}$.

The inverse function. Encryption

$$^{-1} : \mathit{RoleTerm} \to \mathit{RoleTerm}$$

- for any term $rt \in \mathit{RoleTerm}$ we define *the inverse* $rt^{-1} \in \mathit{RoleTerm}$ as follows:

$$rt^{-1} = \begin{cases} sk(t) & \text{if } rt = pk(t) \text{ for some } t \in \mathit{RoleTerm} \\ pk(t) & \text{if } rt = sk(t) \text{ for some } t \in \mathit{RoleTerm} \\ rt & \mathit{otherwise} \end{cases}$$

We further assume that *sk* and *pk* are the secret and, respectively, the public key for asymmetric encryption, while *k* defines a symmetric key.

Example: digital signature

Assume that $m \in \mathit{RoleTerm}$ is a message and $R \in \mathit{Role}$ is a role.

- $\{\!|\, m \,|\!\}_{pk(R)}$ is the encryption of m with the public key of R
- a digitally signed message is a pair

$$(m, \{\!|\, h(m) \,|\!\}_{sk(R)})$$

  where
  - $h(m)$ is *the message digest*, computed using the hash function h,
  - $\{\!|\, h(m) \,|\!\}_{sk(R)}$, the signed message digest, is computed with the signer's private key $sk(R)$,
  - the message in plain text together with the signed message digest form the digitally signed message.
- One can verify the digital signature, in order to prove who the signer was and the integrity of the document:find the message digest using the signer's public key and compare the message digest with the result obtained by applying the hash function to the plain message.

Deduction on *RoleTerm*

$$\vdash \subseteq \mathcal{P}(\mathit{RoleTerm}) \times \mathit{RoleTerm}$$

$M \vdash t$ means that t can be deduced knowing M

$\vdash$ is the least relation with the following properties:

$$\begin{array}{lrll}
\text{if} & t \in M & \text{then} & M \vdash t \\
\text{if} & M \vdash t_1 \text{ and } M \vdash t_2 & \text{then} & M \vdash (t_1, t_2) \\
\text{if} & M \vdash (t_1, t_2) & \text{then} & M \vdash t1 \text{ and } M \vdash t_2 \\
\text{if} & M \vdash t \text{ and } M \vdash k & \text{then} & M \vdash \{\!|\, t \,|\!\}_k \\
\text{if} & M \vdash \{\!|\, t \,|\!\}_k \text{ and } M \vdash k^{-1} & \text{then} & M \vdash t \\
\text{if} & M \vdash t_1 \text{ and } \ldots \text{ and } M \vdash t_n & \text{then} & M \vdash f(t_1, \ldots, t_n)
\end{array}$$

where n is the arity of $f \in \mathit{Func}$.

Notation: $\mathit{Cons}(M) = \{t \in \mathit{RoleTerm} \mid M \vdash t\}$

Exercise: Find $\mathit{Cons}(M)$ where $M = \{\{\!|\, m \,|\!\}_k, \{\!|\, k^{-1} \,|\!\}_{pk(b)}, \{\!|\, h(m) \,|\!\}_m, sk(b)\}$.

Terms: *RoleEvent*

We use *role events* to specify protocol actions and claims.

- Given two disjoint sets:
    - $\mathit{Label} : 1, 2, 3, \ldots$ (labels)
    - $\mathit{Claim} : \mathit{secret}, \mathit{alive}, \ldots$ (denotations for security properties)

  and $R \in \mathit{Role}$ we define

$$
\begin{aligned}
\mathit{RoleEvent}_R \quad ::= \quad & \mathit{send}_{\mathit{Label}}(R, \mathit{Role}, \mathit{RoleTerm}) \\
& \mid \mathit{recv}_{\mathit{Label}}(\mathit{Role}, R, \mathit{RoleTerm}) \\
& \mid \mathit{claim}_{\mathit{Label}}(R, \mathit{Claim}[, \mathit{RoleTerm}])
\end{aligned}
$$

$$\mathit{RoleEvent} = \bigcup_{R \in \mathit{Role}} \mathit{RoleEvent}_R$$

Termeni: *RoleEvent*

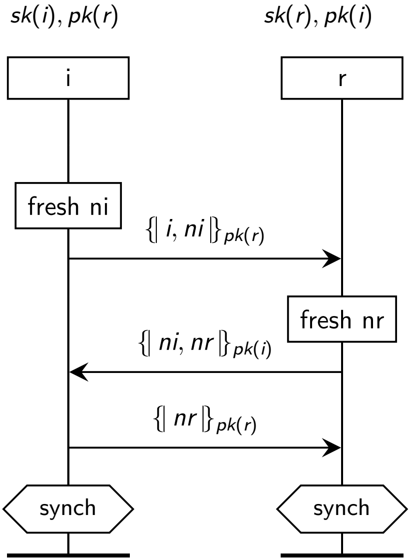
$$
\begin{aligned}
RoleEvent_R \quad ::= \quad & send_{Label}(R, Role, RoleTerm) \\
& \mid recv_{Label}(Role, R, RoleTerm) \\
& \mid claim_{Label}(R, Claim[, RoleTerm])
\end{aligned}
$$

- $send_l(R, R', rt)$ means that R sends the message rt to R',
- $recv_l(R', R, rt)$ means that R receives the message rt (apparently) sent by R',
- $claim_l(R, c, rt)$ is the security property that should be satisfied after the execution of the role R.
- The labels uniquely identify the events and establish the correspondence between *send* and *receive* events.

Protocol specification

Informally, in order to specify a protocol one should describe each role.

$$
\begin{aligned}
NS(i) = \quad & (\{i, r, ni, sk(i), pk(i), pk(r)\}, \\
& [send_1(i, r, \{\!|\, ni, i \,|\!\}_{pk(r)}), \\
& recv_2(r, i, \{\!|\, ni, V \,|\!\}_{pk(i)}), \\
& send_3(i, r, \{\!|\, V \,|\!\}_{pk(r)}), \\
& claim_4(i, synch)])
\end{aligned}
$$



Role description

$$NS(i) = (\{i, r, ni, sk(i), pk(i), pk(r)\}, \\ [send_1(i, r, \{\!|\, ni, i \,|\!\}_{pk(r)}), \\ recv_2(r, i, \{\!|\, ni, V \,|\!\}_{pk(i)}), \\ send_3(i, r, \{\!|\, V \,|\!\}_{pk(r)}), \\ claim_4(i, synch)])$$

- $\{i, r, ni, sk(i), pk(i), sk(r)\}$ is the *initial knowledge* of the role i,
- $s = [send_1(\cdots), \ldots, claim_4(\cdots)]$ is the sequence of events that are executed by i during a protocol session.

Protocol specification

Let P be a protocol and R a role in P.

The specification of the role R in P, denoted $P(R)$, is a pair from $\mathcal{P}(\mathit{RoleTerm}) \times \mathit{RoleEvent}_R^*$.

Example: The roles i and r of NSPK are specified as follows:

$$
\begin{aligned}
NS(i) = \quad & (\{i, r, ni, sk(i), pk(i), pk(r)\}, \\
& [send_1(i, r, \{\!|\, ni, i \,|\!\}_{pk(r)}), \\
& recv_2(r, i, \{\!|\, ni, V \,|\!\}_{pk(i)}), \\
& send_3(i, r, \{\!|\, V \,|\!\}_{pk(r)}), \\
& claim_4(i, synch)])
\end{aligned}
\qquad
\begin{aligned}
NS(r) = \quad & (\{i, r, nr, sk(r), pk(r), pk(i)\}, \\
& [recv_1(i, r, \{\!|\, W, i \,|\!\}_{pk(r)}), \\
& send_2(r, i, \{\!|\, W, nr \,|\!\}_{pk(i)}), \\
& recv_3(i, r, \{\!|\, nr \,|\!\}_{pk(r)}), \\
& claim_5(r, synch)])
\end{aligned}
$$

Thank you!

References

- Cremers, C. J. F. (2006). Scyther : semantics and verification of security protocols Eindhoven: Technische Universiteit Eindhoven DOI: 10.6100/IR614943
- Cremers C. and Mauw S. Operational Semantics and Verification of Security Protocols. Springer, 2012.