Complexity of Anonymity for Security Protocols

Ferucio Laurențiu Țiplea Loredana Vamanu Cosmin Vârlan

Department of Computer Science
"Al.I.Cuza" University of Iași
Iași 700506, Romania
e-mail: {fltiplea,loredana.vamanu,vcosmin}@info.uaic.ro

(ESORICS 2010)

Modeling security protocols

- rules $A \to B : t$ are decomposed into actions:
  - $A!B\ :\ (M)t$ (send action)
  - $B?A\ :\ t$ (receive action)

  (M is the set of all fresh nonces and keys in t).
- security protocol: $\mathcal{P} = (\mathcal{S}, \mathcal{C}, w)$, where:
  - $\mathcal{S}$ is a protocol signature (agents, keys, nonces);
  - $\mathcal{C}$ is the set of protocol constants;
  - w is a sequence of actions;
- role $= w|_A$, where $A \in \mathcal{A}$;
- event $=$ instantiated action;
- state:
  - $s = (s_A | A$ agent$)$;
  - s_A a set of terms (A's knowledge).

A running example

Example

$$
\begin{array}{lcl}
A\,!\,B & : & (\{N_A, K\})\ \{A, B, H, N_A, K\}_{K_B^e} \\
B\,?\,A & : & \{A, B, H, N_A, K\}_{K_B^e} \\
B\,!\,A & : & \{N_A, B, Ticket\}_K, \{N_A, B, Ticket\}_{K_B^d} \\
A\,?\,B & : & \{N_A, B, Ticket\}_K, \{N_A, B, Ticket\}_{K_B^d} \\
A\,!\,C & : & \{Ticket, \{Ticket\}_{K_{AH}}\}_{K_{AC}} \\
C\,?\,A & : & \{Ticket, \{Ticket\}_{K_{AH}}\}_{K_{AC}} \\
C\,!\,H & : & \{\{Ticket\}_{K_{AH}}\}_{K_{CH}} \\
H\,?\,C & : & \{\{Ticket\}_{K_{AH}}\}_{K_{CH}}
\end{array}
$$

Computation rule

We write $s[e\rangle s'$ if and only if:

- if the action of e is $A!B : (M)t$, then:
  - $t \in \overline{s_A \cup M}$ and $M \cap Sub(s) = \emptyset$ (enabling condition)
  - $s'_A = s_A \cup M \cup \{t\}$, $s'_I = s_I \cup \{t\}$, and $s'_C = s_C$, for all $C \in \mathcal{A} - \{A, I\}$;
- if the action of e is $A?B : t$, then:
  - $t \in \overline{s_I}$ (enabling condition)
  - $s'_A = s_A \cup \{t\}$ and $s'_C = s_C$, for all $C \in \mathcal{A} - \{A\}$.

Runs are obtained by interleaving instantiated roles under the enabling condition and preserving the order of events in each role.

Facts

Each action in a security protocol can be described in a logical formalism by using facts of the form $P(t_1, \ldots, t_i)$

Example ($sent$-facts)

Protocol action: $A\,!\,B : (\{N_A, K\})\ \{A, B, H, N_A, K\}_{K_B^e}$
Fact: $sent(A, \{A, B, H, N_A, K\}_{K_B^e}, B)$

Example (rec-facts)

Protocol action: $B\,?\,A : \{A, B, H, N_A, K\}_{K_B^e}$
Facts:

1. passive intruder: $rec(B, \{A, B, H, N_A, K\}_{K_B^e}, A)$;
2. active intruder: $rec(B, \{A, B, H, N_A, K\}_{K_B^e}, (A, I))$.

Facts

Example (gen-facts)

Protocol action: $A\,!\,B : (\{N_A, K\})\ \{A, B, H, N_A, K\}_{K_B^e}$
Fact: $gen(A, \{A, B, H, N_A, K\}_{K_B^e}, B)$
Protocol action: $A\,!\,C : \{Ticket, \{Ticket\}_{K_{AH}}\}_{K_{AC}}$
Fact: $gen(A, \{Ticket\}_{K_{AH}}, H)$

Example ($auth$-facts)

Protocol action: $B\,!\,A : \{N_A, B, Ticket\}_K, \{N_A, B, Ticket\}_{K_B^d}$
Fact: $auth(B, (N_A, B, Ticket, \{N_A, B, Ticket\}_{K_B^d}))$

Augmenting agent states with facts. States

States:

- agent states: $s_A = (s_{A,m}, s_{A,f})$
  - $s_{A,m}$ is a set of messages
  - $s_{A,f}$ is a set of facts
- protocol states: $s = (s_A | A \in \mathcal{A})$

Subterms:

- $Sub(t)$ stands for the set of all subterms of t
- $Sub(s_A)$ stands for $\bigcup_{t \in s_{A,m}} Sub(t)$
- $Sub(s)$ stands for $\bigcup_{A \in \mathcal{A} - \{I\}} Sub(s_A)$

Augmenting agent states with facts. Computation rule

$s[a\rangle s'$, where a is $A!B:(M)t$, if and only if:

1. $t \in \overline{s_{A,m} \cup M}$ and $M \cap Sub(s) = \emptyset$;
2. $s'_{A,m} = s_{A,m} \cup M \cup \{t\}$, $s'_{I,m} = s_{I,m} \cup \{t\}$, and $s'_{C,m} = s_{C,m}$ for any $C \in \mathcal{A} - \{A, I\}$;
3. facts in s' are obtained as follows:
   - 3.1 add $sent(A,t,B)$ to $s_{A,f}$ and $s_{I,f}$;
   - 3.2 add $gen(A,t_1,C)$ to $s_{A,f}$ if $t_1 = \{t'\}_{K_{AC}}$ or $t_1 = \{t'\}_{K^e_C}$ has been built by A in order to build t;
   - 3.3 add $auth(A,t_1)$ to $s_{A,f}$ if $t_1 = (t', \{t'\}_{K^d_A})$ has been built by A in order to build t;
   - 3.4 $s'_{C,f} = s_{C,f}$, for any $C \in \mathcal{A} - \{A, I\}$.

Augmenting agent states with facts. Computation rule

$s[a\rangle s'$, where a is $A?B : t$, if and only if:

1. $t \in \overline{s_{I,m}}$;
2. $s'_{A,m} = s_{A,m} \cup \{t\}$ and $s'_{C,m} = s_{C,m}$, for all $C \in \mathcal{A} - \{A\}$;
3. facts in s' are obtained as follows:
   - 3.1 add $rec(A, t, (B, I))$ to $s_{A,f}$ and $s_{I,f}$;
   - 3.2 $s'_{C,f} = s_{C,f}$, for any $C \in \mathcal{A} - \{A, I\}$.

Passive intruder:

- replace 1 by "$t \in \overline{s_{B,m}}$";
- replace 3.1 by "add $rec(A, t, B)$ to $s_{A,f}$ and $s_{I,f}$".

Fact derivation. Decomposing messages

Definition

A message t is called decomposable over $s = (s_m, s_f)$ if

- $t \in \mathcal{T}_0$, or
- $t = (t_1, t_2)$ for some t_1 and t_2, or
- $t = \{t'\}_K$ for some t' and K with $K^{-1} \in analz(s_m)$, or
- $gen(A, t, B) \in s_f$ for some honest agents A and B.

t is undecomposable over s if $t = \{t'\}_K$ and $K^{-1} \notin analz(s_m)$ and $gen(A, t, B) \notin s_f$, for any A and B honest agents.

Fact derivation. Tracing messages

$trace(t, s)$, where $s = (s_m, s_f)$, outputs the set of all possible messages used to build t:

- $trace(t, s) = \{t\}$, if $t \in \mathcal{T}_0$;
- $trace(t, s) = \{t\} \cup trace(t_1, s) \cup trace(t_2, s)$, if $t = (t_1, t_2)$ for some t_1 and t_2;
- $trace(t, s) = \{t\}$, if t is not decomposable over s;
- $trace(t, s) = \{t\} \cup trace(t', s)$, if $t = \{t'\}_K$ is encrypted but decomposable over s.

Fact derivation. Fact simplification rules

- $(Y1)\ \dfrac{act(A,t,x)}{act(A,t), act(A,x), act(t,x)}$
- $(Y2)\ \dfrac{act(A,t)}{act(A), act(t)}$
- $(Y3)\ \dfrac{act(A,x)}{act(A)}$
- $(Y4)\ \dfrac{act(t,x)}{act(t)}$

where:

- $act = sent \ \Rightarrow \ Y = S \ \wedge \ x = B$, with $B \neq A$;
- $act = rec \ \Rightarrow \ Y = R \wedge \ (x = B \ \vee \ x = (B, I))$, with $B \neq A$.

Fact derivation. Message simplification rules

- $(S5)\ \dfrac{sent(A,t,B),\ t' \in trace(t,s)}{sent(A,t',B)}$
- $(R5)\ \dfrac{rec(A,t,B),\ t' \in trace(t,s)}{rec(A,t',B)}$
- $(R5')\ \dfrac{rec(A,t,(B,I)),\ t' \in trace(t,s)}{rec(A,t',(B,I))}$

where:

- s is an agent state;

Fact derivation

- From rec-facts to gen-facts
  - $(RG)\ \dfrac{rec(B, \{t\}_{K_{AB}})}{gen(A, \{t\}_{K_{AB}}, B)}$
- From rec-facts to $auth$-facts
  - $(RA)\ \dfrac{rec(B, (t, \{t\}_{K_A^d}))}{auth(A, (t, \{t\}_{K_A^d}))}$

Fact derivation. From rec-facts to $sent$-facts

- $(RS1)\ \dfrac{rec(A,t,B)}{sent(B,t,A)}$
- $(RS1')\ \dfrac{rec(A,t,(B,I))}{sent(B)}$
- $(RGS)\ \dfrac{rec(A,t),\ gen(C,t,A)}{sent(C,t,A)}$
- $(RAS)\ \dfrac{rec(A,t),\ auth(C,t)}{sent(C,t)}$

Fact derivation. From rec-facts to rec-facts

- $(RGR)\ \dfrac{rec(A,t,(B,I)),\ gen(B,t,A)}{rec(A,t,B)}$
- $(RAR)\ \dfrac{rec(A,t,(B,I)),\ auth(B,t)}{rec(A,t,B)}$

Fact derivation

- From $sent$-facts to $sent$-facts
  - $(SGS)\ \dfrac{sent(A,t),\ gen(A,t,B)}{sent(A,t,B)}$
- From $sent$-facts to rec-facts
  - $(SGR)\ \dfrac{sent(A,t,B),\ gen(C,t,B)}{rec(A,t,C)}$

Fact derivation: Example of deduction

From (SGR) and $(RS1)$ one can derive:

$$(RGS') \ \frac{rec(A,t,B),\ gen(C,t,A)}{sent(C,t,B)}$$

(RGS') captures a situation like the one in the Kerberos protocol:

$$C \xrightarrow{\{\cdots,\{t\}_{K_{AC}}\}_{K_{BC}}} B \xrightarrow{\{\cdots\},\{t\}_{K_{AC}}} A$$

Observational equivalence over messages

Given a pair of agent states (s, s') define the binary relation $\sim_{s,s'}$ on message terms by:

- $t \sim_{s,s'} t$, for any $t \in \mathcal{T}_0$;
- $t \sim_{s,s'} t'$, for any term t undecomposable over s and any term t' undecomposable over s';
- $(t_1, t_2) \sim_{s,s'} (t'_1, t'_2)$, for any terms t_1, t_2, t'_1, and t'_2 with $t_1 \sim_{s,s'} t'_1$ and $t_2 \sim_{s,s'} t'_2$;
- $\{t\}_K \sim_{s,s'} \{t'\}_K$, for any terms t and t' and any key K with $t \sim_{s,s'} t'$ and $K^{-1} \in analz(s_m) \cap analz(s'_m)$.

Component-wise extend the relation $\sim_{s,s'}$ to facts:

$$P(t_1, \ldots, t_i) \sim_{s,s'} P(t'_1, \ldots, t'_i) \quad \Leftrightarrow \quad (\forall 1 \leq j \leq i)(t_j \sim_{s,s'} t'_j).$$

Observational equivalence over states

- $Analz(M, F)$ = set of all facts that can be inferred from the set F of facts and from the set M of messages;
- $Analz(s) = Analz(s_m, s_f)$, where $s = (s_m, s_f)$.

Definition

Two agent states $s = (s_m, s_f)$ and $s' = (s'_m, s'_f)$ are observationally equivalent, denoted $s \sim s'$, if the following hold:

- $analz(s_m) \cap \mathcal{T}_0 = analz(s'_m) \cap \mathcal{T}_0$;
- for any $\varphi \in Analz(s)$ there is $\varphi' \in Analz(s')$ such that $\varphi \sim_{s,s'} \varphi'$;
- for any $\varphi' \in Analz(s')$ there is $\varphi \in Analz(s)$ such that $\varphi' \sim_{s',s} \varphi$.

Observational equivalence over states

Proposition

1. *The observational equivalence on agent states is an equivalence relation.*
2. *The observational equivalence on agent states is decidable in* $\mathcal{O}(f^4 l^4)$ *time complexity, where* f *is the maximum number of facts in the states, and* l *is the maximum length of the message terms in the states.*

Epistemic logic. Syntax

We use the epistemic logic to reason about anonymity. The syntax of the logic is

$$\varphi ::= p \mid \varphi \wedge \varphi \mid \neg\varphi \mid \mathtt{K}_A\varphi$$

where

- A ranges over a non-empty finite set $\mathcal{A}$ of agent names;
- p ranges over a set Φ of $sent$-, rec-, gen-, and $auth$-facts such that no rec-fact contains terms of the form (B, I).

Epistemic logic. Semantics

The truth value of a formula φ in a security protocol $\mathcal{P}$ is defined inductively as follows:

$$
\begin{array}{lll}
\mathcal{P} \models \varphi & \text{iff} & (\forall s \in Reach(\mathcal{P}))((\mathcal{P}, s) \models \varphi) \\
(\mathcal{P}, s) \models p & \text{iff} & (\exists A \in \mathcal{A} - \{I\})((\mathcal{P}, s_A) \models p) \\
(\mathcal{P}, s_X) \models p & \text{iff} & p \in Analz(s_X) \\
(\mathcal{P}, s) \models \neg\varphi & \text{iff} & (\mathcal{P}, s) \not\models \varphi \\
(\mathcal{P}, s) \models \varphi \wedge \psi & \text{iff} & (\mathcal{P}, s) \models \varphi \wedge\ (\mathcal{P}, s) \models \psi \\
(\mathcal{P}, s) \models \mathtt{K}_A\varphi & \text{iff} & (\forall s' \in Reach(\mathcal{P}))(s' \sim^A s \ \Rightarrow\ (\mathcal{P}, s'_A) \models \varphi)
\end{array}
$$

Anonymity

Anonymity comes in many flavors:

- An Observer does not know that an action has been performed (by an agent);
- An Observer does not know what agent performed an action (group anonymity);
- An Observer does not know what message has been sent/received (message anonymity).

Observer = Honest Agent or the Intruder

Minimal anonymity

Definition

Let $\mathcal{P}$ be a security protocol and X an agent in $\mathcal{P}$ (X may be an honest agent H or the intruder I).
An action act of $\mathcal{P}$ is minimally anonymous w.r.t. X if the following property holds:

$$\mathcal{P} \models act \Rightarrow \neg \mathtt{K}_X act.$$

$$[(\forall s)((s \models act) \Rightarrow (\exists s' \sim^X s)(s'_X \not\models act))]$$

Anonymity of an action which contains messages, such as $sent(A, t)$, should not be confused with the secrecy of t !

Minimal anonymity illustrated

$$
\begin{array}{lcl}
A\,!\,B & : & (\{N_A, K\})\ \{A, B, H, N_A, K\}_{K_B^e} \\
B\,?\,A & : & \{A, B, H, N_A, K\}_{K_B^e} \\
B\,!\,A & : & \{N_A, B, Ticket\}_K, \{N_A, B, Ticket\}_{K_B^d} \\
A\,?\,B & : & \{N_A, B, Ticket\}_K, \{N_A, B, Ticket\}_{K_B^d} \\
A\,!\,C & : & \{Ticket, \{Ticket\}_{K_{AH}}\}_{K_{AC}} \\
C\,?\,A & : & \{Ticket, \{Ticket\}_{K_{AH}}\}_{K_{AC}} \\
C\,!\,H & : & \{\{Ticket\}_{K_{AH}}\}_{K_{CH}} \\
H\,?\,C & : & \{\{Ticket\}_{K_{AH}}\}_{K_{CH}}
\end{array}
$$

- $sent(B, Ticket, A)$ is minimally anonymous w.r.t. C;
- $sent(A, \{Ticket\}_{K_{AH}}, C)$ is not minimally anonymous w.r.t. H because H can learn it by the deduction rule (RGS'), but it is minimally anonymous w.r.t. I because I cannot learn it.

Anonymity decision problems

1. $MAP(\tau)$ – minimal anonymity problem for type τ actions w.r.t. an honest agent

   Instance: security protocol $\mathcal{P}$, action act of type τ, and honest agent H
   Question: is act minimally anonymous w.r.t. H in $\mathcal{P}$?

2. $MAPI(\tau)$ – minimal anonymity problem for type τ actions w.r.t. the intruder

   Instance: security protocol $\mathcal{P}$ and action act of type τ
   Question: is act minimally anonymous w.r.t. I in $\mathcal{P}$?

Undecidability of minimal anonymity

Theorem

$MAP(\tau)$ is undecidable in unrestricted security protocols, for any action type τ.

Theorem

$MAPI(\tau)$ is undecidable in unrestricted security protocols, for any action type τ except for (r, a, a), (r, m, a), and (r, a, m, a).

Proof sketch: simulating CMs by security protocols

Given a counter machine M, we construct a security protocol $\mathcal{P}_M$ as follows:

Initial state:

$$\begin{array}{ccccl} A_0 & ! & B_0 & : & \{z,z\}_K, \{q_0,z,z\}_K, \{z,z\}_K \\ B_0 & ? & A_0 & : & \{z,z\}_K, \{q_0,z,z\}_K, \{z,z\}_K \end{array}$$

A transition $r = (q,0,1,q',1,-1) \in \delta$ is simulated by:

$$\begin{array}{ccccl} A_r & ? & B_r & : & \{z,z\}_K, \{q,z,v\}_K, \{v',v\}_K \\ A_r & ! & B_r & : & (\{u'\})\,\{z,u'\}_K, \{q',u',v'\}_K, \{z,z\}_K \end{array}$$

Proof sketch: simulating CMs by security protocols

Simulating the halting of M:

$$
\begin{array}{lcl}
A_{q_f}\,?\,B_{q_f} & : & \{q_f, u, v\}_K \\
A_{q_f}\,!\,B & : & \{halt, \{halt\}_{K_{A_{q_f}H}}\}_{K_{A_{q_f}B}} \\
B\,?\,A_{q_f} & : & \{halt, \{halt\}_{K_{A_{q_f}H}}\}_{K_{A_{q_f}B}} \\
B\,!\,H & : & \{\{halt\}_{K_{A_{q_f}H}}\}_{K_{BH}} \\
H\,?\,B & : & \{\{halt\}_{K_{A_{q_f}H}}\}_{K_{BH}}
\end{array}
$$

where $halt \in \mathcal{T}_0$ is an initial secret of A_{q_f}.

It is easy to see that M halts iff $sent(A_{q_f}, halt)$ is not minimally anonymous w.r.t. B in $\mathcal{P}$.

Anonymity in bounded security protocols

In bounded security protocol message terms are built over some finite set of basic terms and their length do not exceed some constant k.

for *any reachable state* s *with* $s \models act$ **do**
  if *there exists a reachable state* s' *with* $s' \sim^X s$ *and* $s'_X \not\models act$
  then
    act is minimally anonymous w.r.t. X
  else act is not minimally anonymous w.r.t. X

This algorithm decides minimal anonymity in bounded security protocols but it has a very high complexity.

Anonymity for basic term actions

Definition

An action act of a security protocol is called a *basic-term action* if all terms in the action are basic terms.

For instance, $sent(A, N_A, B)$, where N_A is a nonce, is a basic-term action, whereas $sent(A, \{N_A\}_K, B)$ is not.

Remark

For basic-term actions the following property holds: if $s' \sim^X s$ then $s'_X \not\models act$ if and only if $s_X \not\models act$. Therefore, for basic-term actions, the above algorithm can be simplified by replacing the test in the if-statement by the simpler one "$s_X \not\models act$".

Anonymity for basic term actions

Theorem

$MAP(\tau)$ and $MAPI(\tau)$ are in $NEXPTIME$ for any τ if they are restricted to basic-term actions of type τ and bounded security protocols. Moreover, except for $MAPI(r,a,a)$, $MAPI(r,m,a)$, and $MAPI(r,a,m,a)$, all the other minimal anonymity problems restricted as above are complete for $NEXPTIME$.

Proof sketch

Membership to NEXPTIME:

input : (T,k)-bounded security protocol $\mathcal{P}$, basic-term action act, and agent X;

output: "yes", if act is minimally anonymous w.r.t. X, and "no", otherwise;

begin

  let E be the set of all (T,k)-events of $\mathcal{P}$;

  guess a sequence ξ of pairwise distinct events from E;

  if ξ *is a run of* $\mathcal{P}$ **then**

    let s be the last state of ξ, i.e. $s_0[\xi\rangle s$;

    if $(\mathcal{P},s) \models act \ \wedge \ (\mathcal{P},s_X) \models act$ **then** "no" **else** "yes";

end

Completeness:

Reduce, in polynomial time, any language in $NEXPTIME$ to the complement of each of the problems in the theorem.

Anonymity for basic term actions

If we restrict more bounded security protocols by allowing only 1-session runs, then we obtain the following complexity results.

Theorem

$MAP(\tau)$ and $MAPI(\tau)$ are in NP for any τ if they are restricted to basic-term actions of type τ and 1-session bounded security protocols. Moreover, except for $MAPI(r,a,a)$, $MAPI(r,m,a)$, and $MAPI(r,a,m,a)$, all the other minimal anonymity problems restricted as above are complete for NP.

Proof sketch

Membership to NP:

input : 1-session (T, k)-bounded protocol $\mathcal{P}$, basic-term action act, and agent X;

output: "yes", if act is minimally anonymous w.r.t. X, and "no", otherwise;

begin

  guess a 1-session sequence ξ of $\mathcal{P}$ under some substitution σ;

  if ξ *is a run of* $\mathcal{P}$ **then**

    let s with $s_0[\xi\rangle s$;

    if $(\mathcal{P}, s) \models act \ \wedge \ (\mathcal{P}, s_X) \models act$ **then** "no" **else** "yes";

end

Completeness:

Reduce, in polynomial time, 3-SAT to the complement of each of the problems in the theorem.

Conclusions

We have:

1. proposed a formalization of (minimal) anonymity in security protocols;
2. shown that minimal anonymity is undecidable in unrestricted security protocols;
3. established the complexity of minimal anonymity for basic term actions in bounded security protocols.

Unreported work:

1. group anonymity in security protocols;
2. unlinkability in security protocols;
3. relationship and complexity results for minimal anonymity, group anonymity, and unlinkability.