Formal verification of secured routing protocols

Mathilde Arnaud, Véronique Cortier, Stéphanie Delaune

LSV, ENS Cachan & CNRS & INRIA Saclay Île de France

LORIA, CNRS & INRIA Nancy Grand Est, France
Ad Hoc Networks

- Networks with little or no infrastructure
- Open infrastructure
- Agents can only communicate directly with their immediate neighbors
Propagation of a message in the network
Propagation of a message in the network
Propagation of a message in the network
Propagation of a message in the network
Routing Protocols

**Protocol** series of rules describing how each participant should behave in order to achieve a common goal

**Routing** goal: allowing distant nodes to communicate

Specificities of **routing** protocols:
- broadcast communication
- importance of the topology of the network
- number of agents involved unknown
Example: simplified DSR

Request phase: A wants to speak to D
Example: simplified DSR

Reply phase: routes found are sent back to A

- route: [A; B; C; D]
- route: [A; E; B; C; D]
What if E was dishonest?

\[A; E; D\]

\[A\]

\[E\]

\[B\]

\[C\]

\[D\]

→ A would get a false route to D!

⇒ Securing routing protocols
Insecure network: traditional description

Presence of an attacker

- may read every message sent on the net,
- may intercept and send new messages.
Results for cryptographic protocols

Attacks against protocols

Existence of an attack is **decidable** (for a bounded number of sessions).

Tools have been conceived that automatically detect logical flaws: Proverif, AVISPA, Scyther...

**but** not applicable to **routing protocols** because of their specificities.
Secure Routing Protocols

Goal: allowing distant nodes to communicate by finding a path between them while guaranteeing security

Notable differences with other cryptographic protocols:
- broadcast communication and topology of the network
- specific tests and security properties (e.g. route correctness)
- a form of recursivity
A Dolev-Yao intruder controls the network:
- hears all messages
- chooses which messages to transmit
- does not follow the protocol

Power of the intruder in an ad hoc setting:
- **broadcast** → cannot delete messages
- **located** ↔ cannot hear distant messages
Our goals

Modeling and analysing secured routing protocols, taking into account:

- network topology
- less powerful intruder
- tests on the topology
- recursivity?
Messages are abstracted by terms

Agents: $a, b, \ldots$

Keys: $k_1, k_2, \ldots$

Concatenation: $\langle m_1, m_2 \rangle$

Lists: $[]$, $a :: l$

Encryption: $\{m\}_k$

Signature: $\lbrack m \rbrack_k$

Example: The message $\lbrack \langle A, K_a \rangle \rbrack_K$ is represented by:

```
  [ ]
  /    \
|     |
\langle \rangle K
  /    \
  /    |
A     K_a
```

Intuition: only the structure of the message is kept.
Intruder abilities

Composition rules

\[
\frac{u_1 \quad u_2}{\langle u_1, u_2 \rangle} \quad \frac{u_1 \quad u_2}{u_1 :: u_2} \quad \frac{u_1 \quad sk(u_2)}{\llbracket u_1 \rrbracket_{sk(u_2)}} \quad \frac{u_1 \quad u_2}{\{ u_1 \}_u_2}
\]
Intruder abilities

Composition rules

\[ \frac{u_1 \quad u_2}{\langle u_1, u_2 \rangle} \]
\[ \frac{u_1 \quad u_2}{u_1 :: u_2} \]
\[ \frac{u_1 \quad \text{sk}(u_2)}{[u_1]_{\text{sk}(u_2)}} \]
\[ \frac{u_1 \quad u_2}{\{u_1\} u_2} \]

Decomposition rules

\[ \frac{\langle u_1, u_2 \rangle}{u_i \quad i \in \{1,2\}} \]
\[ \frac{u_1 :: u_2}{u_i \quad i \in \{1,2\}} \]
\[ \frac{\{u_1\} u_2 \quad u_2}{u_1} \]

Optional rule:
\[ \frac{[u_1]_{\text{sk}(u_2)}}{u_1} \]
Intruder abilities

Composition rules

\[
\begin{align*}
\frac{u_1 \quad u_2}{\langle u_1, u_2 \rangle} & & \frac{u_1 \quad u_2}{u_1 :: u_2} & & \frac{u_1 \quad \text{sk}(u_2)}{\llbracket u_1 \rrbracket_{\text{sk}(u_2)}} & & \frac{u_1 \quad u_2}{\{ u_1 \}_{u_2}}
\end{align*}
\]

Decomposition rules

\[
\begin{align*}
\frac{\langle u_1, u_2 \rangle}{u_i} & & i \in \{1, 2\} & & \frac{u_1 :: u_2}{u_i} & & i \in \{1, 2\} & & \frac{\{ u_1 \}_{u_2} \quad u_2}{u_1}
\end{align*}
\]

Optional rule:

\[
\frac{\llbracket u_1 \rrbracket_{\text{sk}(u_2)}}{u_1}
\]

Deducibility relation

A term \( u \) is **deducible** from a set of terms \( T \), denoted by \( T \vdash u \), if there exists a prooftree witnessing this fact.
Inspired from CBS#, introduced by Nanz and Hankin

\[ P, Q ::= \]
\[ 0 \]
\[ \text{out}(u).P \]
\[ \text{in } u[\Phi].P \]
\[ \text{store}(u).P \]
\[ \text{read } u \text{ then } P \text{ else } Q \]
\[ \text{if } \Phi \text{ then } P \text{ else } Q \]
\[ P \mid Q \]
\[ !P \]
\[ \text{new } m.P \]

Processes
null process
emission
reception, \( \Phi \in \mathcal{L} \)
storage
reading
conditional, \( \Phi \in \mathcal{L} \)
parallel composition
replication
fresh name generation

State: \( [P]_n, [S]_n, \mathcal{I} \)
Formulas

\[ \Phi ::= \]
\[ \text{check}(a, b) \]
\[ \text{checkl}(c, l) \]
\[ \text{loop}(l) \]
\[ \text{route}(l) \]
\[ \Phi_1 \land \Phi_2 \]
\[ \Phi_1 \lor \Phi_2 \]
\[ \neg \Phi \]

Property and tests expressed in \( \mathcal{L} \)

Formula

- \( a \) and \( b \) are neighbors
- \( l \) is locally correct for \( c \)
- existence of a loop in a list
- validity of a route
- conjunction
- disjunction
- negation
Expressiveness of the model

Concerning the specificities of routing protocols:

- list as a data structure
- broadcast communication and network topology
- specific tests and security properties
Example: source node

\[ S : \text{out}(u_1).\text{in } u_2[\Phi_S] \]
Example: source node

\[ S : \text{out}(u_1).\text{in} u_2[\Phi_S] \rightarrow \text{in} u_2[\Phi_S] \]
Example: intermediate node

\[ W : \text{in } w_1[\Phi_W].\text{store}(t).\text{out}(w_2).0 \]
Example: intermediate node

\[ \sigma = \text{mgu}(u_1, w_1) \quad \text{and} \quad w = w_2\sigma \]
Concrete transitions

Problem with concrete transitions: infinitely many possibilities

\[
\begin{align*}
\left[ \text{in } x. \text{out}(x) \right]_n & \rightarrow \left[ \text{out}(t_1) \right]_n \quad \text{if } I \vdash t_1 \\
& \rightarrow \left[ \text{out}(t_2) \right]_n \quad \text{if } I \vdash t_2 \\
& \rightarrow \left[ \text{out}(t_3) \right]_n \quad \text{if } I \vdash t_3 \\
& \vdots \quad \vdots
\end{align*}
\]

→ Introduction of symbolic transitions to avoid state explosion by keeping some variables
Symbolic transition: Example

Concrete transition

\[
\text{Concrete transition}
\]

\[
\begin{align*}
\text{Concrete transition} & \quad \text{Symbolic transition} \\
[\text{in } u[\Phi].P]_n \cup \mathcal{P}; S; \mathcal{I} & \quad \text{Symbolic transition} \\
& \quad \rightarrow \\
& \quad [P\sigma]_n \cup \mathcal{P}; S; \mathcal{I} \\
& \quad \text{if } \mathcal{I} \vdash t, [\Phi\sigma] = 1 \\
& \quad (n_I, n) \in E, \\
& \quad \text{and } \sigma = \text{mgu}(t, u) \\
& \quad \text{if } (n_I, n) \in E
\end{align*}
\]

All possible concrete transitions are captured in one symbolic transition
Secrecy via constraint solving [Millen et al]

Constraint systems are used to specify secrecy preservation under a particular, finite scenario.

\[
\begin{align*}
\text{Scenario} & \quad \text{Constraint System} \\
\text{rcv}(u_1) \xrightarrow{N_1} \text{snd}(v_1) & \quad T_0 \vdash u_1 \\
\text{rcv}(u_2) \xrightarrow{N_2} \text{snd}(v_2) & \quad T_0, v_1 \vdash u_2 \\
\ldots & \quad \ldots \\
\text{rcv}(u_n) \xrightarrow{N_n} \text{snd}(v_n) & \quad T_0, v_1, \ldots, v_n \vdash s
\end{align*}
\]

where \( T_0 \) is the initial knowledge of the attacker.

**Remark:** Constraint Systems may be used more generally for trace-based properties, e.g. authentication.
Secrecy via constraint solving [Millen et al]

Constraint systems are used to specify secrecy preservation under a particular, finite scenario.

**Scenario**

\[
\text{rcv}(u_1) \xrightarrow{N_1} \text{snd}(v_1)
\]

\[
\text{rcv}(u_2) \xrightarrow{N_2} \text{snd}(v_2)
\]

\[\ldots\]

\[
\text{rcv}(u_n) \xrightarrow{N_n} \text{snd}(v_n)
\]

**Constraint System**

\[
C = \begin{cases} 
T_0 \vdash u_1 \\
T_0, v_1 \vdash u_2 \\
\ldots \\
T_0, v_1, \ldots, v_n \vdash s
\end{cases}
\]

**Solution of a constraint system**

A substitution $\sigma$ such that for every $T \vdash u \in C$, $u\sigma$ is deducible from $T\sigma$, that is $T\sigma \vdash u\sigma$. 
How do we show decidability?

Step 1. Simplifying the constraint system
→ common step to all our results

Step 2. Bounding solutions
→ specific techniques for
1. routing protocols with topology tests
2. protocols with recursivity
Goal of the simplification: obtain simpler constraint systems

Definition (Solved constraint systems)

Solved constraint systems are of the form
\[ C = T_1 \vdash x_1 \land \cdots \land T_n \vdash x_n, \text{ where the } x_i \text{ are variables.} \]

We show that we can simplify constraint system and obtain solutions to solved constraint systems.
Simplification rules

\[ R_{ax} : \quad C \land T \vdash u \quad \leadsto \quad C \quad \text{if } T \cup \{ x \mid T' \vdash x \in C, T' \subsetneq T \} \vdash u \]

\[ R_{unif} : \quad C \land T \vdash u \quad \leadsto_{\sigma} \quad C\sigma \land T\sigma \vdash u\sigma \quad \text{if } \sigma = \text{mgu}(t_1, t_2) \text{ where } t_1, t_2 \in \text{st}(T, u), \text{ and } t_1 \neq t_2 \]

\[ R_{fail} : \quad C \land T \vdash u \quad \leadsto \quad \bot \quad \text{if } \text{vars}(T, u) = \emptyset \text{ and } T \not\vdash u \]

\[ R_f : \quad C \land T \vdash f(u, v) \quad \leadsto \quad C \land T \vdash u \land T \vdash v \]
**Theorem**

\( C \) has a solution iff \( C \xrightarrow{\sigma}^* C' \) with \( C' \) in solved form.

**New characterization property**

if \( T_i\theta \vdash u \), we have that \( T_i\theta \vdash u \) using composition rules.

\( T_1, \ldots, T_n \) represent a basis for deducible terms.
We show decidability in two cases:

- decidability of an attack for a given topology
- existence of a topology leading to an attack
Ad Hoc Routing Protocols
Modeling routing protocols

Fixed topology

Theorem

Let $P$ be a protocol without replication and $\Phi$ a property. Deciding whether there is an attack on $P$ and $\Phi$ for a given topology is NP-complete.

1. guess a path of symbolic execution
2. reduce to solved constraint systems (extension of the approach of Millen, Shmatikov and Comon-Lundh with an infinity of names and extended signature)
3. decide existence of a solution
Existence of a topology leading to an attack

**Theorem**

Let $P$ be a protocol and $\Phi$ a property. Deciding whether there exists a topology such that there is an attack on $P$ and $\Phi$ is NP-complete.

- Guess the edges between the nodes appearing in the protocol
- As in the case of a fixed topology, reduce the problem to solving a constraint system in solved form
- Bounding the size of the solution (in the size of the initial configuration) allows to bound the size of the graph
Protocols with recursivity tests

Aim: Analysing protocols that involve iterative or recursive operations.

- group protocols
- certification paths for public keys
- delegation rights
- secured source routing protocols
Example 1: Certificate Chains

Public keys need to be certified

Example (X.509 public key certificates)

\[ [ [ \langle A_1, \text{pub}(A_1) \rangle ]_{sk(A_2)} ; [ \langle A_2, \text{pub}(A_2) \rangle ]_{sk(A_3)} ; \ldots ; [ \langle A_{n-1}, \text{pub}(A_{n-1}) \rangle ]_{sk(A_n)} ; [ \langle A_n, \text{pub}(A_n) \rangle ]_{sk(S)} ] \]

where

- \( S \) is some trusted server, and
- each agent \( A_{i+1} \) certifies the public key \( \text{pub}(A_i) \) of agent \( A_i \).
Example 2: Secured Source Routing

To certify the route, each node signs the fact that it belongs to it.

Example (SMNDP)

The route is represented by $l_{route} = [A_n; \ldots; A_1]$. The expected message is of the form

$[[\langle A_n, A_0, l_{route} \rangle]_{sk(A_1)}; [\langle A_n, A_0, l_{route} \rangle]_{sk(A_2)}; \ldots$ $\ldots; [\langle A_n, A_0, l_{route} \rangle]_{sk(A_n)}].$

Remark: $[\langle A_n, A_0, l_{route} \rangle]_{sk(A_i)}$ both depends on the list $l_{route}$ and on its $i$-th element.
Our decidability results

We prove decidability for constraint systems with recursive tests:
- for link-based recursive languages $\mathcal{L}_{link}$

**Example (Certificate chains)**

$$\mathcal{L}_1 = \{\llbracket \langle A_1, pub(A_1) \rangle \rrbracket_{sk(A_2)}; \llbracket \langle A_2, pub(A_2) \rangle \rrbracket_{sk(A_3)}; \ldots \ldots; \llbracket \langle A_n, pub(A_n) \rangle \rrbracket_{sk(S)} \mid A_1, \ldots, A_n \ \text{agent names}, \ n \in \mathbb{N} \}$$

- for mapping-based languages $\mathcal{L}_{mapping}$

**Example (SMNDP)**

$$\mathcal{L}_2 = \llbracket \langle A_n, A_0, l_{route} \rangle \rrbracket_{sk(A_1)}; \llbracket \langle A_n, A_0, l_{route} \rangle \rrbracket_{sk(A_2)}; \ldots \ldots; \llbracket \langle A_n, A_0, l_{route} \rangle \rrbracket_{sk(A_n)} \mid l_{route} = [A_n, \ldots, A_1], \ n \in \mathbb{N} \}$$
Case of link-based recursive language

Links are terms containing variables that can be instantiated by basic terms, e.g. names.

Chains in such a language are lists of links recursively constrained:

- **Pattern**: Represents the basic structure.
- **Valid Chain**: Shows an example of a chain that matches the pattern.
Encoding the example

Certificate lists are all built from the term \( m = \llbracket \langle x, \text{pub}(y) \rangle \rrbracket_{sk(z)} \)

\[
\llbracket \langle x, \text{pub}(x) \rangle \rrbracket_{sk(y)} :: \llbracket \langle y, \text{pub}(y) \rangle \rrbracket_{sk(z)} :: \ldots \llbracket \langle w, \text{pub}(w) \rangle \rrbracket_{sk(s)}
\]

The basic certificate chain is of the form \( \llbracket \langle w, \text{pub}(w) \rangle \rrbracket_{sk(s)} \)
Theorem

Let $\mathcal{L}$ be a link-based recursive language. Let $\mathcal{C}$ be a constraint system and $\phi$ be a conjunction of $\mathcal{L}$-language constraints. Deciding whether $\mathcal{C}$ and $\phi$ has a solution is in NP.

Intuitively, bounding the solution is done by limiting the number of possible links
Case of mapping-based languages

From a base shape $b$ and a list of names $\ell = [a_0; \ldots, a_n]$, the following terms can be built:

- $b_0 = b[a_0, \bot]$
- $b_1 = b[a_1, [b_0]]$
- $b_2 = b[a_2, [b_1; b_0]]$
  
  :  

- $b_n = b[a_n, [b_{n-1}; \ldots; b_0]]$

$(\ell, \ell') \in \mathcal{L}$ if and only if $\ell' = [b_n; b_{n-1} \ldots; b_0]$
Theorem

Let $\mathcal{L}$ be a mapping-based recursive language. Let $\mathcal{C}$ be a constraint system and $\phi$ be a conjunction of $\mathcal{L}$-language constraints. Deciding whether $\mathcal{C} \land \phi$ has a solution is in NP.

Intuition to bound the lists: the beginning of $\ell = [a_0; \ldots; a_n]$ constrains the end of $\ell' = [b_n; \ldots; b_0]$ and reciprocally.

Idea: cut in the middle
Conclusion

- Model for ad hoc networks
- Decidability for routing protocols for fixed and for unknown topologies
- NP decision procedures for security protocols with recursive tests for two classes of tests

Future work:
- Full analysis of recursive routing protocols
- Implementation
- Anonymous routing?