02244 Language-Based Security
Security Protocols
Automated Analysis II: Abstraction

Sebastian Mödersheim

April 23, 2018

The Sources of Infinity

- For security protocols, the state space can be infinite for (at least) the following reasons:

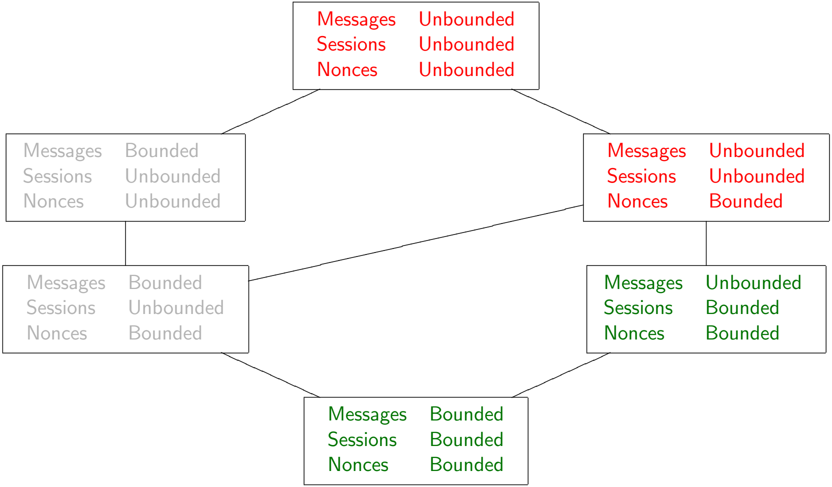
  Messages The intruder can compose arbitrarily complex messages from his knowledge, e.g. $i, h(i), h(h(i)), \ldots$.

  Sessions No bound on the number of executions of the protocol. (In our model: infinitely many threads in the initial state).

  Nonces In an unbounded number of sessions, honest agents create an infinite number of fresh nonces.

- Consider the models that arise from bounding any subset of these parameters:
  - ★ Decidability/Automation?
  - ★ Can we justify the bounds?

Decidability Lattice



Today we look at the remaining two elements.

Typed Model

Declare for all variables and constants a type, e.g.,

$A, B, C, a, b, c, i, \ldots$	*Agent*
$NA, NB, na_{17}, \ldots$	*Nonce*
$KAB, kab, \ldots$	*SymmetricKey*

Typed Model

We allow that variables are only instantiated with constants of the same type.

- This means bounding the depth of messages. Why?

Typed Model

Declare for all variables and constants a type, e.g.,

$A, B, C, a, b, c, i, \ldots$	*Agent*
$NA, NB, na_{17}, \ldots$	*Nonce*
$KAB, kab, \ldots$	*SymmetricKey*

Typed Model

We allow that variables are only instantiated with constants of the same type.

- This means bounding the depth of messages. Why?
- Can this be justified?
  - ★ i.e., don't we exclude some attacks with this in general?

No Fresh Nonces

Idea

Consider a scenario where agents do not create fresh nonces in every protocol run, but use the same nonce in all protocol runs with the same communication partners.

Example NSPK

Replace the fresh nonces with functions of the involved agent names. For NSPK, we choose NA becomes $na(A, B)$ and NB becomes $nb(B, A)$.

$$
\begin{array}{llll}
A & \to & B & : & \{na(A, B), A\}_{pk(B)} \\
B & \to & A & : & \{na(A, B), nb(B, A)\}_{pk(A)} \\
A & \to & B & : & \{nb(B, A)\}_{pk(B)} \\
\hline
\multicolumn{5}{l}{NA \ \texttt{secret of} \ A, B} \\
\multicolumn{5}{l}{NB \ \texttt{secret of} \ A, B}
\end{array}
$$

No Fresh Nonces

Idea

Consider a scenario where agents do not create fresh nonces in every protocol run, but use the same nonce in all protocol runs with the same communication partners.

Example NSPK:

Role A: replace NA with $na(A,B)$.

$$\bullet \xrightarrow{\{na(A,B),A\}_{\mathsf{pk}(B)}}$$

$$\bullet \xleftarrow{\{na(A,B),NB\}_{pk(A)}}$$

$$\bullet \xrightarrow{\{NB\}_{pk(B)}}$$

Leave received nonce NB as is.

Role B: replace NB with $nb(B,A)$.

$$\xrightarrow{\{NA,A\}_{\mathsf{pk}(B)}} \bullet$$

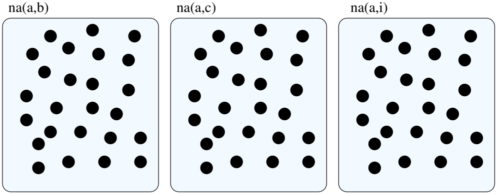
$$\xleftarrow{\{NA,nb(B,A)\}_{pk(A)}} \bullet$$

$$\xrightarrow{\{nb(B,A)\}_{pk(B)}} \bullet$$

Leave received nonce NA as is.

Abstract Interpretation

We have partitioned the set of all concrete nonces
into abstract equivalence classes:

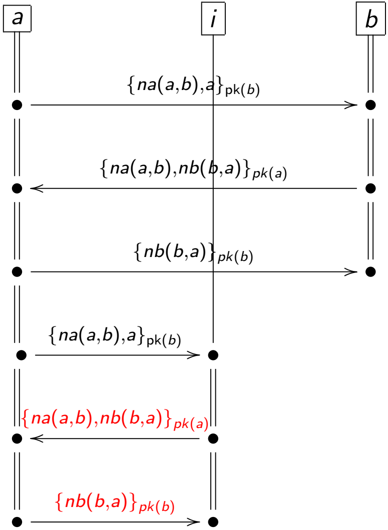


a's nonces for communication with b, c, and i

and replace in the protocol model each concrete value
with its abstract equivalence class.

Abstract Interpretation

- Every reachable concrete state has an abstract counter-part,
- but some abstract states have no concrete counter-part:

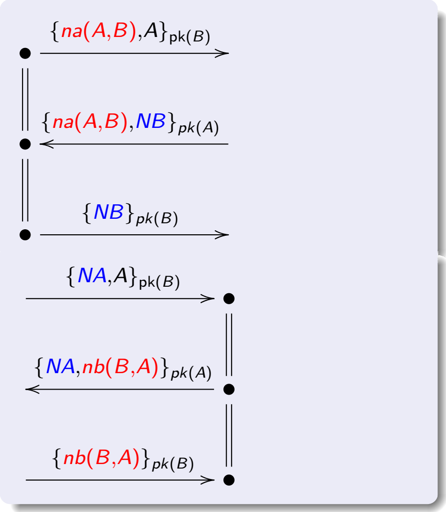


Abstract Interpretation

- Every reachable concrete state has an abstract counter-part,
- but some abstract states have no concrete counter-part.
- So the abstract model is an over-approximation of the concrete model, allowing at least as many behaviors.
- The abstract model is easier to verify than the concrete one.
- Claim: if the abstract model has no attack trace, then the concrete model has none. (True?)
- An attack trace in the abstract model may be a false positive: no (corresponding) attack trace in the concrete model.

Abstracting States

Idea: regard honest agents just as a set of oracles that the intruder can ask to get an answer from.



$$ik(\{na(A,B),A\}_{pk(B)})$$

$$ik(\{na(A,B),NB\}_{pk(A)}) \implies ik(\{NB\}_{pk(B)})$$

$ik(m)$: intruder knows message m

$$ik(\{NA,A\}_{pk(B)}) \implies ik(\{NA,nb(B,A)\}_{pk(A)})$$

(Intruder learns nothing)

Abstracting States

There is no notion of states or development anymore. Rather we consider with $ik(m)$ all messages that the intruder can ever know.

$\mathrm{ik}(\{na(A,B),A\}_{\mathrm{pk}(B)})$ for all agents A, B

$\mathrm{ik}(\{NA,A\}_{\mathrm{pk}(B)}) \Rightarrow \mathrm{ik}(\{NA,nb(B,A)\}_{\mathrm{pk}(A)})$

$\mathrm{ik}(\{na(A,B),NB\}_{\mathrm{pk}(A)}) \Rightarrow \mathrm{ik}(\{NB\}_{\mathrm{pk}(B)})$

…plus standard intruder deduction rule (Dolev-Yao):

$\mathrm{ik}(M) \wedge \mathrm{ik}(K) \Rightarrow \mathrm{ik}(\{M\}_K)$

$\mathrm{ik}(\{M\}_K) \wedge \mathrm{ik}(\mathrm{inv}(K)) \Rightarrow \mathrm{ik}(M)$

$\mathrm{ik}(M_1) \wedge \mathrm{ik}(M_2) \Rightarrow \mathrm{ik}(\langle M_1, M_2\rangle)$

$\mathrm{ik}(\langle M_1, M_2\rangle) \Rightarrow \mathrm{ik}(M_1) \wedge \mathrm{ik}(M_2)$

Initial intruder knowledge: $\mathrm{ik}(\mathrm{inv}(\mathrm{pk}(i)))$ $\mathrm{ik}(A)$ $\mathrm{ik}(pk(A))$ for every A

Goal: For every honest $A \neq i$ and $B \neq i$:

$\mathrm{ik}(na(A,B)) \Rightarrow attack$

$\mathrm{ik}(nb(B,A)) \Rightarrow attack$

Abstracting States

There is no notion of states or development anymore. Rather we consider with $ik(m)$ all messages that the intruder can ever know.

$\mathrm{ik}(\{na(A,B),A\}_{\mathrm{pk}(B)})$ for all agents A, B

$\mathrm{ik}(\{NA,A\}_{\mathrm{pk}(B)}) \Rightarrow \mathrm{ik}(\{NA,nb(B,A)\}_{\mathrm{pk}(A)})$

$\mathrm{ik}(\{na(A,B),NB\}_{\mathrm{pk}(A)}) \Rightarrow \mathrm{ik}(\{NB\}_{\mathrm{pk}(B)})$

...plus standard intruder deduction rule (Dolev-Yao):

$\mathrm{ik}(M) \wedge \mathrm{ik}(K) \Rightarrow \mathrm{ik}(\{M\}_K)$

$\mathrm{ik}(\{M\}_K) \wedge \mathrm{ik}(\mathrm{inv}(K)) \Rightarrow \mathrm{ik}(M)$

$\mathrm{ik}(M_1) \wedge \mathrm{ik}(M_2) \Rightarrow \mathrm{ik}(\langle M_1, M_2\rangle)$

$\mathrm{ik}(\langle M_1, M_2\rangle) \Rightarrow \mathrm{ik}(M_1) \wedge \mathrm{ik}(M_2)$

Initial intruder knowledge: $\mathrm{ik}(\mathrm{inv}(\mathrm{pk}(i)))$ $\mathrm{ik}(A)$ $\mathrm{ik}(pk(A))$ for every A

Goal: For every honest $A \neq i$ and $B \neq i$:

$\mathrm{ik}(na(A,B)) \Rightarrow attack$

$\mathrm{ik}(nb(B,A)) \Rightarrow attack$

Can you write this in Prolog?

The Fixedpoint for NSPK

1. $\{na(A, B), A\}_{pk(B)}$ by step 1

The Fixedpoint for NSPK

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1 when $B = i$

The Fixedpoint for NSPK

1. $\{na(A, B), A\}_{pk(B)}$ by step 1
2. $na(A, i)$ by intruder deduction on 1 when $B = i$
3. $\{na(A, B), nb(B, A)\}_{pk(A)}$ by step 2 on 1

The Fixedpoint for NSPK

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1 when $B = i$
3. $\{na(A,B),nb(B,A)\}_{pk(A)}$ by step 2 on 1
4. $\{N,nb(B,A)\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint

The Fixedpoint for NSPK

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1 when $B = i$
3. $\{na(A,B),nb(B,A)\}_{pk(A)}$ by step 2 on 1
4. $\{N,nb(B,A)\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint
   (e.g. $N = na(A,i)$ by 2)

The Fixedpoint for NSPK

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1 when $B = i$
3. $\{na(A,B),nb(B,A)\}_{pk(A)}$ by step 2 on 1
4. $\{N,nb(B,A)\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint
   (e.g. $N = na(A,i)$ by 2)
5. $nb(B,i)$ by intruder deduction on 4

The Fixedpoint for NSPK

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1 when $B = i$
3. $\{na(A,B),nb(B,A)\}_{pk(A)}$ by step 2 on 1
4. $\{N,nb(B,A)\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint
   (e.g. $N = na(A,i)$ by 2)
5. $nb(B,i)$ by intruder deduction on 4
6. $\{nb(B,A)\}_{pk(B)}$ by step 3 on 3

The Fixedpoint for NSPK

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1 when $B=i$
3. $\{na(A,B),nb(B,A)\}_{pk(A)}$ by step 2 on 1
4. $\{N,nb(B,A)\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint
   (e.g. $N=na(A,i)$ by 2)
5. $nb(B,i)$ by intruder deduction on 4
6. $\{nb(B,A)\}_{pk(B)}$ by step 3 on 3
7. $nb(i,A)$ by intruder deduction on 6

The Fixedpoint for NSPK

1. $\{na(A, B), A\}_{pk(B)}$ by step 1
2. $na(A, i)$ by intruder deduction on 1 when $B = i$
3. $\{na(A, B), nb(B, A)\}_{pk(A)}$ by step 2 on 1
4. $\{N, nb(B, A)\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint (e.g. $N = na(A, i)$ by 2)
5. $nb(B, i)$ by intruder deduction on 4
6. $\{nb(B, A)\}_{pk(B)}$ by step 3 on 3
7. $nb(i, A)$ by intruder deduction on 6
8. $\{N\}_{pk(i)}$ by step 3 on 2 and intruder ded.

The Fixedpoint for NSPK

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1 when $B=i$
3. $\{na(A,B),nb(B,A)\}_{pk(A)}$ by step 2 on 1
4. $\{N,nb(B,A)\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint
   (e.g. $N = na(A,i)$ by 2)
5. $nb(B,i)$ by intruder deduction on 4
6. $\{nb(B,A)\}_{pk(B)}$ by step 3 on 3
7. $nb(i,A)$ by intruder deduction on 6
8. $\{N\}_{pk(i)}$ by step 3 on 2 and intruder ded.
9. $\{nb(B,A)\}_{pk(i)}$ by step 3 on 4 with $N = na(A,i)$

The Fixedpoint for NSPK

1. $\{na(A, B), A\}_{pk(B)}$ by step 1
2. $na(A, i)$ by intruder deduction on 1 when $B = i$
3. $\{na(A, B), nb(B, A)\}_{pk(A)}$ by step 2 on 1
4. $\{N, nb(B, A)\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint (e.g. $N = na(A, i)$ by 2)
5. $nb(B, i)$ by intruder deduction on 4
6. $\{nb(B, A)\}_{pk(B)}$ by step 3 on 3
7. $nb(i, A)$ by intruder deduction on 6
8. $\{N\}_{pk(i)}$ by step 3 on 2 and intruder ded.
9. $\{nb(B, A)\}_{pk(i)}$ by step 3 on 4 with $N = na(A, i)$
10. $nb(B, A)$ intruder deduction on 9 — attack

Note on Variables

We have left variables A, B, and N in the above fixedpoint:

- A and B represent arbitrary agent names
- N represents any nonce the intruder knows

Note on Variables

We have left variables A, B, and N in the above fixedpoint:

- A and B represent arbitrary agent names
- N represents any nonce the intruder knows

- In general, this represents an infinite fixedpoint, when replacing arbitrary agent names for A and B like $a1, a2, a3, \ldots$

Note on Variables

We have left variables A, B, and N in the above fixedpoint:

- A and B represent arbitrary agent names
- N represents any nonce the intruder knows

- In general, this represents an infinite fixedpoint, when replacing arbitrary agent names for A and B like $a1, a2, a3, \ldots$
- There are some results that show: for many problems it is sufficient to work with only a fixed number of agents $\{a, b, i\}$. But makes the *descriptions* of the fixedpoint longer...

Note on Variables

- We can easily calculate with facts that have variables, but mind incorrect variable capturing:
  - ★ Fact $\{na(A, i), nb(B, A)\}_{pk(A)}$ of the previous fixedpoint
  - ★ and the rule $\{na(A, B), NB\}_{pk(A)} \Rightarrow \{NB\}_{pk(B)}$.

Note on Variables

- We can easily calculate with facts that have variables, but mind incorrect variable capturing:
  - ★ Fact $\{na(A,i), nb(B,A)\}_{pk(A)}$ of the previous fixedpoint
  - ★ and the rule $\{na(A,B), NB\}_{pk(A)} \Rightarrow \{NB\}_{pk(B)}$.
  - ★ Seems to allow only $B = i$ and $NB = nb(i,A)$?

Note on Variables

- We can easily calculate with facts that have variables, but mind incorrect variable capturing:
  - ★ Fact $\{na(A,i), nb(B,A)\}_{pk(A)}$ of the previous fixedpoint
  - ★ and the rule $\{na(A,B), NB\}_{pk(A)} \Rightarrow \{NB\}_{pk(B)}$.
  - ★ Seems to allow only $B = i$ and $NB = nb(i,A)$?
  - ★ But A and B can be anybody, so we could rename this fact to:

$$\{na(A',i), nb(B',A')\}_{pk(A')}$$

Note on Variables

- We can easily calculate with facts that have variables, but mind incorrect variable capturing:
  - Fact $\{na(A,i), nb(B,A)\}_{pk(A)}$ of the previous fixedpoint
  - and the rule $\{na(A,B), NB\}_{pk(A)} \Rightarrow \{NB\}_{pk(B)}$.
  - Seems to allow only $B = i$ and $NB = nb(i,A)$?
  - But A and B can be anybody, so we could rename this fact to:

$$\{na(A',i), nb(B',A')\}_{pk(A')}$$

  - Now solution: $A = A'$, $B = i$, $NB = nb(B',A)$ more general!

Note on Variables

- We can easily calculate with facts that have variables, but mind incorrect variable capturing:
  - ★ Fact $\{na(A,i), nb(B,A)\}_{pk(A)}$ of the previous fixedpoint
  - ★ and the rule $\{na(A,B), NB\}_{pk(A)} \Rightarrow \{NB\}_{pk(B)}$.
  - ★ Seems to allow only $B = i$ and $NB = nb(i,A)$?
  - ★ But A and B can be anybody, so we could rename this fact to:

$$\{na(A',i), nb(B',A')\}_{pk(A')}$$

  - ★ Now solution: $A = A'$, $B = i$, $NB = nb(B',A)$ more general!
  - ★ Thus the intruder obtains the secret nonce between any two agents B' and A. (attack)

Lowe's fix for NSPK (NSL)

Insert the name of B in the second message:

Example (NSL)

$$\begin{array}{lcll} A & \rightarrow & B: & \{NA, A\}_{\mathsf{pk}(B)} \\ B & \rightarrow & A: & \{NA, NB, B\}_{\mathsf{pk}(A)} \\ A & \rightarrow & B: & \{NB\}_{\mathsf{pk}(B)} \end{array}$$

The Fixedpoint for NSL

1. $\{na(A, B), A\}_{pk(B)}$ by step 1

The Fixedpoint for NSL

1. $\{na(A, B), A\}_{pk(B)}$ by step 1
2. $na(A, i)$ by intruder deduction on 1

The Fixedpoint for NSL

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1
3. $\{na(A,B),nb(B,A),B\}_{pk(A)}$ by step 2 on 1

The Fixedpoint for NSL

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1
3. $\{na(A,B),nb(B,A),B\}_{pk(A)}$ by step 2 on 1
4. $\{N,nb(B,A),B\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint
   (e.g. $N = na(A,i)$ by 2)

The Fixedpoint for NSL

1. $\{na(A,B), A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1
3. $\{na(A,B), nb(B,A), B\}_{pk(A)}$ by step 2 on 1
4. $\{N, nb(B,A), B\}_{pk(A)}$ by step 2 with intr. ded. for any nonce N that the intruder can derive in this fixedpoint (e.g. $N = na(A,i)$ by 2)
5. $nb(B,i)$ by intruder deduction on 3

The Fixedpoint for NSL

1. $\{na(A,B), A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on ➊
3. $\{na(A,B), nb(B,A), B\}_{pk(A)}$ by step 2 on ➊
4. $\{N, nb(B,A), B\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint
   (e.g. $N = na(A,i)$ by ➋)
5. $nb(B,i)$ by intruder deduction on ➌
6. $\{nb(B,A)\}_{pk(B)}$ by step 3 on ➌

The Fixedpoint for NSL

1. $\{na(A, B), A\}_{pk(B)}$ by step 1
2. $na(A, i)$ by intruder deduction on 1
3. $\{na(A, B), nb(B, A), B\}_{pk(A)}$ by step 2 on 1
4. $\{N, nb(B, A), B\}_{pk(A)}$ by step 2 with intr. ded. for any nonce N that the intruder can derive in this fixedpoint (e.g. $N = na(A, i)$ by 2)
5. $nb(B, i)$ by intruder deduction on 3
6. $\{nb(B, A)\}_{pk(B)}$ by step 3 on 3
7. $nb(i, A)$ by intruder deduction on 6

The Fixedpoint for NSL

1. $\{na(A,B), A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1
3. $\{na(A,B), nb(B,A), B\}_{pk(A)}$ by step 2 on 1
4. $\{N, nb(B,A), B\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint
   (e.g. $N = na(A,i)$ by 2)
5. $nb(B,i)$ by intruder deduction on 3
6. $\{nb(B,A)\}_{pk(B)}$ by step 3 on 3
7. $nb(i,A)$ by intruder deduction on 6
8. $\{N\}_{pk(i)}$ by step 3 on 2 and intruder ded.

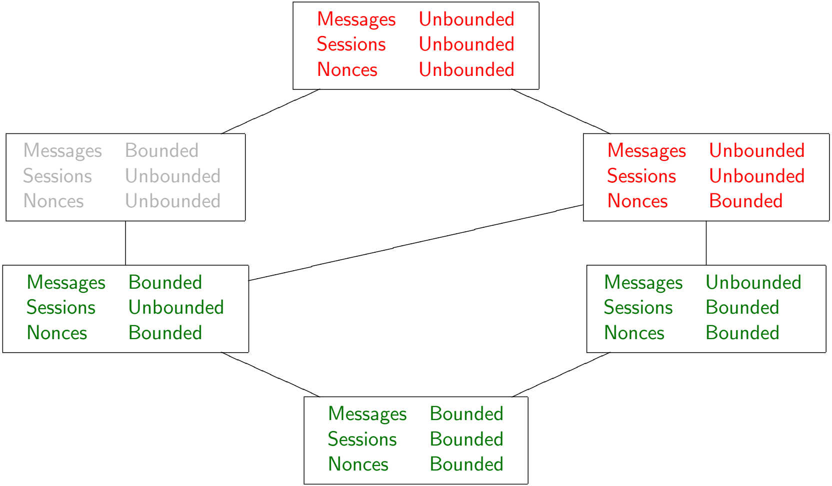
The Fixedpoint for NSL

1. $\{na(A,B),A\}_{pk(B)}$ by step 1
2. $na(A,i)$ by intruder deduction on 1
3. $\{na(A,B),nb(B,A),B\}_{pk(A)}$ by step 2 on 1
4. $\{N,nb(B,A),B\}_{pk(A)}$ by step 2 with intr. ded.
   for any nonce N that the intruder can derive in this fixedpoint
   (e.g. $N = na(A,i)$ by 2)
5. $nb(B,i)$ by intruder deduction on 3
6. $\{nb(B,A)\}_{pk(B)}$ by step 3 on 3
7. $nb(i,A)$ by intruder deduction on 6
8. $\{N\}_{pk(i)}$ by step 3 on 2 and intruder ded.
   Not derivable anymore: $\{nb(B,A)\}_{pk(i)}$

Abstraction Based Analysis

- We have now a verification procedure for unbounded sessions when bounding fresh nonces and messages...
- This also avoids the entire state explosion problem of standard model-checking: the number of reachable states is (at least) exponential in the number of concurrent processes.

Decidability Lattice



ProVerif

- ProVerif [Blanchet 2001ff] is a protocol verifier based on the abstract-interpretation method where messages are unbounded by default.
- Horn clauses like $P_1 \wedge \ldots \wedge P_n \Rightarrow Q$ can be equivalently written as $\neg P_1 \vee \ldots \vee \neg P_n \vee Q$.

Resolution

Find two clauses $p \vee \phi$ and $\neg p \vee \psi$. Then we can derive the clause $\phi \vee \psi$

- Add the clause ¬attack: if the protocol has an attack, then this leads to a contradiction, deriving the empty clause "False".
- Resolution is refutation-complete: a set of clauses is consistent iff False is not derivable with resultion.
- With bounded messages, resolution is guaranteed to terminate.
- In an untyped model without bounding messages, this approach can lead to non-termination, but often it does terminate!

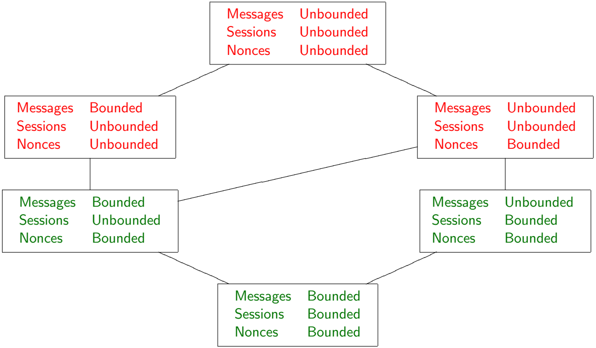
Decidability

- To complete our picture:

Theorem ([Durgin et al. 2004])

For an unbounded number of sessions and an unbounded number of nonces, protocol security is undecidable, even when bounding messages.

Decidability Lattice



Conclusion:

For decidability, we may have either unbounded messages or unbounded sessions (with bounded nonces), but not both.

Bibliography

- ProVerif: http://prosecco.gforge.inria.fr/personal/bblanche/proverif/
- Sebastian Mödersheim and Georgios Katsoris. *A Sound Abstraction of the Parsing Problem*. CSF 2014.
- Hubert Comon and Véronique Cortier. *Security Properties: Two Agents Are Sufficient*. In ESOP 2003.