Term Rewriting applied to Cryptographic Protocol Analysis: the Maude-NPA tool

Santiago Escobar

Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València
sescobar@dsic.upv.es
Outline

1. Formal Analysis of Protocols
   - The Needham-Schroeder Public Key
   - Motivating Protocols
   - Some Examples of Algebraic Identities

2. Introduction to Rewriting Logic

3. How Maude-NPA works

4. Examples of execution
Outline

1. Formal Analysis of Protocols
   - The Needham-Schroeder Public Key
   - Motivating Protocols
   - Some Examples of Algebraic Identities

2. Introduction to Rewriting Logic

3. How Maude-NPA works

4. Examples of execution
Formal Analysis of Protocols

- Crypto protocol analysis in the **standard model** is well understood.
- Need to support algebraic properties of some protocols
  - Diffie-Hellman exponentiation,
  - exclusive-or,
  - homomorphism (one-sided distributivity)
- These operations well understood in the bounded sessions case
  - Decidability results for exclusive-or, exponentiation, homomorphisms, etc.
- What is lacking:
  1. more **general understanding**, especially for unbounded sessions,
  2. **tool support**.
Our approach

- Use **rewriting logic** as general theoretical framework
  - protocols and intruder rules specified as *transition rewrite rules*
  - crypto properties as *oriented equational properties and axioms*
- Use narrowing modulo equational theories in two ways
  - as a *symbolic reachability analysis method*
  - as an *extensible equational unification method*
- Combine with state reduction techniques (grammars, optimizations, etc.)
- Implement in **Maude** programming environment
  - Rewriting logic gives us *theoretical framework* and understanding
  - Maude implementation gives us *tool support*
Our Plans

1. Start by formalizing NPA techniques in rewriting logic (2005)
2. Extend model to different types of equational theories (2006)
   • Explicit Encryption and Decryption, AC-unification, Diffie-Hellman Exponentiation, Exclusive-or
4. Document and distribute the tool (v1.0 2007)
6. Integrate dedicated unification algorithms (2011)
   • Homomorphism, Exclusive-or
7. Document and distribute the tool (v2.0 2012)
8. Extensive protocol analysis (2012-now)
   • Homomorphism, Exclusive-or, Abelian groups
9. Advanced properties:
   • Indistinguishability (2013-now), Conditional protocols (2016)
10. Standard APIs: IBM CCA, PKCS#11, Yubikey (2014-now)
11. Document and distribute the tool (v3.0 2016)
Outline

1. Formal Analysis of Protocols
   The Needham-Schroeder Public Key
   Motivating Protocols
   Some Examples of Algebraic Identities
Building Blocks for Security Protocols

**Cryptographic Procedures:** encryption of messages.

\[ \{\{ M \} K_B \} K_B^{-1} = M \]

**(Pseudo-)Random Number Generators:** to generate “nonces”, e.g. for “challenge/response”.

**Protocols:** recipe for exchanging messages.

Steps like: *A sends B her name together with the message M. The pair \( \{ A, M \} \) is encrypted with B’s public key.*

\[ A \rightarrow B : \{ A, M \} K_B \]
An authentication protocol

The Needham-Schroeder Public Key protocol (NSPK):

1. \(A \rightarrow B: \{NA, A\}_{KB}\)
2. \(B \rightarrow A: \{NA, NB\}_{KA}\)
3. \(A \rightarrow B: \{NB\}_{KB}\)

Goal: mutual authentication. Translation:

\[\begin{align*}
\{NA, A\}_{KB} & \quad \text{“This is Alice and I have chosen a nonce } NA.\text{”} \\
\{NA, NB\}_{KA} & \quad \text{“Here is your nonce } NA. \text{ Since I could read it, I must be Bob. I also have a challenge } NB \text{ for you.”} \\
\{NB\}_{KB} & \quad \text{“You sent me } NB. \text{ Since only Alice can read this and I sent it back, you must be Alice.”}
\end{align*}\]

NSPK proposed in 1970s and used for decades, until...

Protocols are typically small and convincing... and often wrong!
How to at least tie against a Chess Grandmaster
Man-in-the-middle attack on NSPK

B believes he is speaking with A!
What went wrong?

- Problem in step 2:

\[ B \mapsto A : \{N_A, N_B\}_{K_A} \]

- Agent B should also give his name: NA, NB, BKA.
- The improved version is called \textbf{NSL protocol} by Gavin Lowe.
- Is the protocol now correct?
Needham-Schroeder-Lowe Public Key Exchange Protocol

A aborts the protocol execution!
(or ignores the message)
Outline

1 Formal Analysis of Protocols
   The Needham-Schroeder Public Key
   Motivating Protocols
   Some Examples of Algebraic Identities
Example: Needham-Schroeder Public Key Protocol

**Protocol (text-book)**

\[ A \rightarrow B : pk(B,A;N_A) \]
\[ B \rightarrow A : pk(A,N_A;N_B) \]
\[ A \rightarrow B : pk(B,N_B) \]

**Attack sequence**

1. \( (pk(i, a; n(a, r1)))^{+} \)
2. \( (pk(i, n(b, r2)))^{-} \)
3. \( (a; n(a, r1))^{+} \)
4. \( (a; n(a, r1))^{-} \)
5. \( (pk(b, a; n(a, r1)))^{+} \)
6. \( (pk(b, a; n(a, r1)))^{-} \)
7. \( (pk(a, n(a, r1); n(b, r2)))^{+} \)
8. \( (pk(a, n(a, r1); n(b, r2)))^{-} \)
9. \( (pk(i, n(b, r2)))^{+} \)
10. \( (pk(i, n(b, r2)))^{-} \)
11. \( (n(b, r2))^{+} \)
12. \( (n(b, r2))^{-} \)
13. \( (pk(b, n(b, r2)))^{+} \)
14. \( (pk(b, n(b, r2)))^{-} \)
Example: Needham-Schroeder-Lowe Protocol

**Protocol (text-book)**

\[ \begin{align*}
A & \rightarrow B : pk(B, A; N_A) \\
B & \rightarrow A : pk(A, N_A; N_B; B) \\
A & \rightarrow B : pk(B, N_B)
\end{align*} \]
Example: NSL-xor Protocol

**Protocol (text-book)**

\[
A \rightarrow B : pk(B, A; N_A) \\
B \rightarrow A : pk(A, N_A; N_B \oplus B) \\
A \rightarrow B : pk(B, N_B)
\]

**Attack sequence**

1. \((pk(i, a; n(a, r1)))^+\)
2. \((pk(i, n(b, r2)))^-\)
3. \((a; n(a, r1))^+\)
4. \((a; n(a, r1))^-\)
5. \((pk(b, a; n(a, r1)))^+\)
6. \(generatedByIntruder(b \oplus i)\)
7. \((pk(b, a; n(a, r1)))^-\)
8. \((pk(a, n(a, r1); n(b, r2); b))^+\)
9. \((pk(a, n(a, r1); n(b, r2); b))^-\)
10. \((pk(i, n(b, r2) \oplus b \oplus i))^+\)
11. \((pk(i, n(b, r2) \oplus b \oplus i))^-\)
12. \((n(b, r2) \oplus b \oplus i)^+\)
13. \((b \oplus i)^-\)
14. \((n(b, r2) \oplus b \oplus i)^+\)
15. \((n(b, r2))^+\)
16. \((n(b, r2))^-\)
17. \((pk(b, n(b, r2)))^+\)
18. \((pk(b, n(b, r2)))^-\)
Example: NSL-homomorphism Protocol

Protocol (text-book)

\[ A \rightarrow B : \text{pk}(B, A; N_A) \]
\[ B \rightarrow A : \text{pk}(A, N_A; N_B; B) \]
\[ A \rightarrow B : \text{pk}(B, N_B) \]

Attack sequence

1. \( \text{generatedByIntruder}(\text{pk}(a, i)) \)
2. \( \text{generatedByIntruder}(\text{pk}(b, a; NI)) \)
3. \( (\text{pk}(b, a; NI))^− \)
4. \( (\text{pk}(a, NI; n(b, r2); b))^+ \)
5. \( (\text{pk}(a, NI); \text{pk}(a, n(b, r2)); \text{pk}(a, b))^− \)
6. \( (\text{pk}(a, n(b, r2)); \text{pk}(a, b))^+ \)
7. \( (\text{pk}(a, n(b, r2)); \text{pk}(a, b))^− \)
8. \( (\text{pk}(a, n(b, r2)))^+ \)
9. \( (\text{pk}(a, i))^− \)
10. \( (\text{pk}(a, n(b, r2)))^− \)
11. \( (\text{pk}(i, a); \text{pk}(a, n(b, r2)))^+ \)
12. \( \text{pk}(a, i; n(b, r2))^− \)
13. \( (\text{pk}(i, n(b, r1); n(a, r1); a))^+ \)
14. \( (\text{pk}(i, n(b, r2)); \text{pk}(i, n(a, r1)); \text{pk}(i, a))^− \)
15. \( (\text{pk}(i, n(b, r2)))^+ \)
16. \( (\text{pk}(i, n(b, r2)))^− \)
17. \( (n(b, r2))^+ \)
18. \( (n(b, r2))^− \)
19. \( (\text{pk}(b, n(b, r2)))^+ \)
20. \( (\text{pk}(b, n(b, r2)))^− \)
Outline

1 Formal Analysis of Protocols
   The Needham-Schroeder Public Key
   Motivating Protocols
   Some Examples of Algebraic Identities
Explicit Encryption and Decryption

• Most formal models lack explicit decryption operator and assume implicit decryption
• If a principal knows an encrypted message and the key, assume principal can decrypt message under the following conditions
  • Implicit assumption that principal never decrypts a message that wasn’t encrypted with a key known by the principal
  • Assumption that principals can check format of decrypted message
• What if these assumptions do not hold?
• In that case, need to model both encryption and decryption symbols explicitly, plus their cancellation, e.g. \( d(K, e(K, Y)) = Y \).

**Example:** Needham-Schroeder Public Key (NSPK)
Modular Exponentiation in Diffie-Hellman

- Basic DH example protocol (each nonzero residue mod $P$ is a power of $g$)
  1. $A \rightarrow B : g^{NA} \mod P$
     - $B$ computes $(g^{NA})^{NB} \mod P$
  2. $B \rightarrow A : g^{NB} \mod P$
     - $A$ and $B$ compute $(g^{NB})^{NA} = (g^{NA})^{NB} \mod P$ and get a shared secret key.

- Properties:

  $$(g^X)^Y = g^{X*Y} = g^{Y*X} = (g^Y)^X$$
  $$(X * Y) * Z = X * (Y * Z) \quad X * Y = Y * X$$

  of modular exponentiation in order to faithfully represent this protocol

**Example:** Diffie-Hellman Protocol
Exclusive-Or

- Cheap and has provable security properties
  - If we send $X \oplus R$, where $R$ a random secret, observer learns no more about $X$ than before it saw message
- On the other hand, associativity-commutativity and cancellation properties make it tricky to reason about

\[
\begin{align*}
X \oplus Y &= Y \oplus X \\
(X \oplus Y) \oplus Z &= X \oplus (Y \oplus Z) \\
X \oplus X &= 0 \\
X \oplus 0 &= X
\end{align*}
\]

**Example:** Needham-Schroeder-Lowe with XOR (NSL-xor)
Homomorphism

- The electronic codebook (ECB) encryption splits a message into blocks and cyphers the blocks using the same key.

Identical plaintext blocks are encrypted into identical ciphertext blocks (does not hide data patterns well). Sensitive to the property:

\[ e(K, X; Y) = e(K, X) \cdot e(K, Y) \]

**Example:** NSL with homomorphic encryption
Outline

1 Formal Analysis of Protocols
   The Needham-Schroeder Public Key
   Motivating Protocols
   Some Examples of Algebraic Identities

2 Introduction to Rewriting Logic

3 How Maude-NPA works

4 Examples of execution
Definition

A rewrite theory \( \mathcal{R} \) is a triple \( \mathcal{R} = (\Sigma, E, R) \), with:

- \( (\Sigma, R) \) a set of rewrite rules of the form \( t \rightarrow s \)
  e.g. \( e(K, N_A; X) \rightarrow e(K, X) \)

- \( (\Sigma, E) \) a set of equations of the form \( t = s \)
  e.g. \( d(K, e(K, Y)) = Y \)

Intuitively, \( \mathcal{R} \) specifies a concurrent system, whose states are elements of the initial algebra \( T_{\Sigma/E} \) specified by \( (\Sigma, E) \), and whose concurrent transitions are specified by the rules \( R \).

\[
e(k, n_a; m) \in T_{\Sigma/E}
\]
\[
d(k_2, e(k_2, e(k, n_a; m))) \notin T_{\Sigma/E}
\]
Rewriting modulo

Definition

Given \((\Sigma, E, R)\), \(t \rightarrow_{R,E} s\) if there is

- a position \(p \in \text{Pos}(t)\);
- a rule \(l \rightarrow r\) in \(R\);
- a matching \(\sigma\) (modulo \(E\))

such that \(t|_p =_E \sigma(l)\), and \(s = t[\sigma(r)]_p\).

Example:

- \(R = \{ \ e(K,N_A;X) \rightarrow e(K,X) \ \} \)
- \(E = \{ \ d(K,e(K,Y)) = Y \ \} \)
- \(e(k, n_A;m) \rightarrow_{R,E} e(k,m) \)
- \(d(k, e(k,e(k_2,n_A;m))) =_E e(k_2,n_a;m) \rightarrow_{R,E} e(k_2,m) \)
Narrowing and Backwards Narrowing

**Definition**

Given \((\Sigma, E, R)\), \(t \stackrel{\sigma,R,E}{\sim} s\) if there is
- a non-variable position \(p \in \text{Pos}(t)\);
- a rule \(l \rightarrow r \in R\);
- a unifier \(\sigma\) (modulo \(E\)) such that \(\sigma(t|_p) =_E \sigma(l)\), and \(s = \sigma(t[r]_p)\).

**Example:**
- \(R = \{ e(K,N_A;X) \rightarrow e(K,X) \} \)
- \(E = \{ d(K,e(K,Y)) = Y \} \)
- \(e(k, X) \stackrel{\{X \mapsto N_A; X\}', R,E}{\sim} e(k, X')\)
- \(d(k, X) \stackrel{\{X \mapsto e(k,e(K,N_A;X'))\}', R,E}{\sim} e(K, X')\)

**Backwards Narrowing:** Narrowing with rewrite rules reversed
Narrowing can be used as a general deductive procedure for solving symbolic reachability problems of the form

\[(\exists \vec{x}) \ t_1(\vec{x}) \rightarrow t'_1(\vec{x}) \land \ldots \land t_n(\vec{x}) \rightarrow t'_n(\vec{x})\]

in a given rewrite theory.

- The terms \(t_i\) and \(t'_i\) denote sets of states (all the possible instances of the term)
- Symbolyc reachability means for what subset of states denoted by \(t_i\) are the states denoted by \(t'_i\) reachable?
- No finiteness assumptions about the state space.
Equational Unification

Definition

Given an order-sorted equational theory \((\Sigma, Ax \uplus E)\) and \(t ? t'\), an \((Ax \uplus E)\)-unifier is an order-sorted subst. \(\sigma\) s.t. \(\sigma(t) =_{Ax \uplus E} \sigma(t')\).

Compared to syntactic unification:

- \(f(a, X) = f(Y, b)\) has solution \(X \mapsto b, Y \mapsto a\)
- \(f(a, X) =_{AC} f(b, Y)\) has solution \(X \mapsto b, Y \mapsto a\)
- \(X + 0 =_{ACU} X\), where 0 is the identity, has solution \(id\)
- \(X + a + b =_{XOR} a\) has solution \(X \mapsto b, Y \mapsto a\)
Equational Unification - Complete

When $Ax = \emptyset$ and $E$ convergent TRS

Narrowing provides a complete (but semi-decidable) $E$-unification procedure [Hullot80]. e.g. cancellation $d(K,e(K,M)) \rightarrow M$.

When $Ax \neq \emptyset$ and $E$ convergent and coherent TRS modulo $Ax$

Narrowing provides a complete (but semi-decidable) $E$-unification procedure [Jouannaud-Kirchner-Kirchner-83] e.g. exclusive-or

$X * 0 \rightarrow X, X * X \rightarrow 0 \mid (X * Y) * Z = X * (Y * Z), X * Y = Y * X$
Equational Unification - Decidable

When $Ax = \emptyset$

**Basic narrowing** strategy [Hullot80] is **complete** for normalized substitutions.
Cases where basic narrowing terminates have been studied [Alpuente-Escobar-Iborra-TCS09].

When $Ax \neq \emptyset$

**Folding variant-narrowing** [Escobar-Meseguer-Sasse-JLAP12] is the most promising strategy for equational unification. **Fully implemented in Maude.**
**$E,Ax$-variants**

**$E,Ax$-variant**

Given a term $t$ and an equational theory $Ax \cup E$, $(t', \theta)$ is an $E,Ax$-variant of $t$ if $\theta(t) \downarrow_{E,Ax} = Ax t'$ [Comon-Delaune-RTA05]

**Finite and complete set of $E,Ax$-variants**

$\forall \sigma \text{ s.t. } \sigma(t) \downarrow_{E,Ax} = t'$, $\exists (t'', \theta) \in V_{E,Ax}(t)$ s.t.

1. $t''$ is in $\rightarrow_{E,Ax}$-normal form
2. $t'$ and $t''$ ($\sigma \downarrow_{E,Ax}$ and $\theta$) are just renamings modulo $Ax$.

**Finite Variant Property**

Theory has FVP if there is a finite number of most general $E,Ax$-variants for every term.
$E,Ax$-variants - Example

\[
\begin{align*}
X \oplus 0 &\rightarrow X \\
X \oplus X &\rightarrow 0 \\
X \oplus X \oplus Y &\rightarrow Y
\end{align*}
\]

(cancellation rules: $E$)

\[
\begin{align*}
X \oplus (Y \oplus Z) & = (X \oplus Y) \oplus Z \\
X \oplus Y & = Y \oplus X
\end{align*}
\]

(axioms: $Ax$)

- For $X \oplus X$ only $E,Ax$-variant is: $(0, id)$
- For $X \oplus Y$ there are 7 most general $E,Ax$-variants
  1. $(X \oplus Y, id)$
  2. $(0, \{X \mapsto U, Y \mapsto U\})$
  3. $(Z, \{X \mapsto 0, Y \mapsto Z\})$
  4. $(Z, \{X \mapsto Z \oplus U, Y \mapsto U\})$
  5. $(Z, \{X \mapsto Z, Y \mapsto 0\})$
  6. $(Z, \{X \mapsto U, Y \mapsto Z \oplus U\})$
  7. $(Z_1 \oplus Z_2, \{X \mapsto U \oplus Z_1, Y \mapsto U \oplus Z_2\})$
Narrowing & Unification in Maude-NPA

- Cryptographic protocols are modeled as a rewrite theory \( \mathcal{P} = (\Sigma, \Delta \uplus B, R) \)
- Narrowing at two levels in Maude-NPA
  1. A theory \((\Sigma, \Delta \uplus B, R)\): \(\Delta \uplus B\)-narrowing with rules \(R\)
  2. For \(\Delta \uplus B\)-unification \((B\)-narrowing with rules \(\Delta\))
- \(\Delta \uplus B\)-unification for each backwards step using \(R\)
  1. Built-in Maude ACU unification algorithms
  2. Dedicated unification algorithms (xor, homomorphism)
  3. Hybrid approach: built-in algorithms for \(B\), and a generic algorithm (variant narrowing) for \(\Delta\).
Outline

1. Formal Analysis of Protocols
   - The Needham-Schroeder Public Key
   - Motivating Protocols
   - Some Examples of Algebraic Identities

2. Introduction to Rewriting Logic

3. How Maude-NPA works

4. Examples of execution
Maude-NPA

- A tool to find or prove the absence of attacks
- Analyzes infinite state systems:
  - Active Dolev-Yao intruder
  - No abstraction or approximation of nonces
  - Unbounded number of sessions
- Performs symbolic backwards search from an insecure state to find attacks or to prove unreachability of cryptographic protocols
- Sensitive to past and future
Basic Structure of Maude-NPA

- Honest principal and intruder actions are modeled as a strand space (Thayer, Herzog, and Guttman)
Basic Structure of Maude-NPA

- **A strand** is a sequence of positive and negative terms
  - **Negative term** stand for received message
  - **Positive terms** stand for sent messages
  - **Example:**
    
    (honest) \[ pke(B, N_A; A)^+, pke(A, N_A; N_B)^-, pke(B, N_B)^+ \]
    
    (intruder \[ X^-, pke(A, X)^+ \] and \[ X^-, Y^-, (X; Y)^+ \])

- Modified strand notation: a marker denoting the current state
  - **Example:** \[ pke(B, N_A; A)^+ | pke(A, N_A; N_B)^-, pke(B, N_B)^+ \]

- Strand annotated with fresh terms generated by principal executing strands (to obtain an infinite number of nonces)
  
  \[ \triangleright r \triangleright [ pke(B, n(A, r); A)^+ | pke(A, n(A, r); N_B)^-, pke(B, N_B)^+ ] \]

- **Intruder knowledge** explicitly represented
  - \( m \in \mathcal{I} \): terms already learnt by the intruder
  - \( m \notin \mathcal{I} \): terms the intruder does not know, but that will be learnt
Basic Structure of Maude-NPA

- A state is a set of strands plus the intruder knowledge

\[ \ldots [\ nil, m_1^\pm, \ldots, m_i^\pm \ | \ m_{i+1}^\pm, \ldots, m_k^\pm, \ nil ] \ & \ \{ t_1 \notin \mathcal{I}, \ldots, t_j \notin \mathcal{I}\}, \{ s_1 \in \mathcal{I}, \ldots, s_m \in \mathcal{I}\} \]

- Initial strand \[ [\ nil \ | \ m_1^\pm, \ldots, m_n^\pm, \ nil ] \]
- Final strand \[ [\ nil, m_1^\pm, \ldots, m_n^\pm, \ | \ nil ] \]
- Initial Intruder knowledge \( \{ t_1 \notin \mathcal{I}, \ldots, t_n \notin \mathcal{I}\} \)
- Final Intruder knowledge \( \{ t_1 \in \mathcal{I}, \ldots, t_n \in \mathcal{I}\} \)
Protocol Rules and Their Execution

\[
[\text{\texttt{nil},m_{1}^{\pm},\ldots,m_{i}^{\pm} \mid m_{i+1}^{\pm},\ldots,m_{k}^{\pm},\text{\texttt{nil}}}] \& \\
\{t_{1} \notin \mathcal{I},\ldots,t_{j} \notin \mathcal{I}\}, \{s_{1} \in \mathcal{I},\ldots,s_{m} \in \mathcal{I}\}
\]

- Negative message \( m_{i}^{-} \) in the past part of the strand is
  - \( E \)-unified with a term already known by the intruder \( s_{p} \in \mathcal{I} \)
  - or introduced into the intruder knowledge as \( m_{i} \in \mathcal{I} \)
- Positive message \( m_{i}^{+} \) in the past part of the strand is
  - \( E \)-unified with term known by the intruder \( s_{p} \in \mathcal{I} \), and then \( s_{p} \notin \mathcal{I} \) is transformed into \( s_{p} \in \mathcal{I} \)

\[
\begin{array}{c}
\text{\texttt{m} \notin \mathcal{I}} \\
\text{\texttt{m} \in \mathcal{I}}
\end{array}
\]
Protocol Rules and Their Execution

To execute a protocol $\mathcal{P}$ associate to it a rewrite theory on sets of strands as follows. Let $\mathcal{I}$ informally denote the set of terms known by the intruder, and $K$ the facts known or unknown by the intruder

1. $[L | M^-, L'] \& \{M \in \mathcal{I}, K\} \rightarrow [L, M^- | L'] \& \{M \in \mathcal{I}, K\}$
   Moves input messages into the past

2. $[L | M^+, L'] \& \{K\} \rightarrow [L, M^+ | L'] \& \{K\}$
   Moves output message that are not read into the past

3. $[L | M^+, L'] \& \{M \notin \mathcal{I}, K\} \rightarrow [L, M^+ | L'] \& \{M \in \mathcal{I}, K\}$
   Joins output message with term in intruder knowledge.

For backwards execution, just reverse
Introducing New Strands

- If we want an **unbounded number of strands**, need some way of introducing new strands in the backwards search
- Specialize rule 3 using each strand of the protocol \( \mathcal{P} \):

\[
\{ [ l_1 | u^+, l_2 ] \} \& \{ u \notin \mathcal{I}, K \} \rightarrow \{ u \in \mathcal{I}, K \}
\]

s.t. \( [ l_1, u^+, l_2 ] \in \mathcal{P} \)
Backwards Reachability Analysis

- Backwards narrowing protocol execution defines a backwards reachability relation
- Specify a state describing the **attack state**, including a set of final strands plus terms $u \in \mathcal{I}$ and $u \not\in \mathcal{I}$
- Execute the protocol backwards to an **initial state**, if possible
- In initial step, prove lemmas that identify certain states unreachable (if necessary)
- For each intermediate state found, several **optimizations** available (check if it can be proved unreachable and discard)
- Also **global optimizations** (super lazy intruder, state subsumption)
Outline

1 Formal Analysis of Protocols
   The Needham-Schroeder Public Key
   Motivating Protocols
   Some Examples of Algebraic Identities

2 Introduction to Rewriting Logic

3 How Maude-NPA works

4 Examples of execution
Example: Needham-Schroeder Public Key Protocol

**Protocol (text-book)**

\[ A \rightarrow B : \text{pk}(B,A;N_A) \]
\[ B \rightarrow A : \text{pk}(A,N_A;N_B) \]
\[ A \rightarrow B : \text{pk}(B,N_B) \]

**Protocol (strand spaces)**

\[ :: \ r1 :: [\text{nil} | (\text{pk}(B,A;n(A,r1)))^+, \]
\[ (pk(A,n(A,r1);N_B))^-, \]
\[ pk(B,N_B)^+] \]
\[ :: \ r2 :: [\text{nil} | (pk(B,A;N_A))^-., \]
\[ (pk(A,N_A;n(B,r2)))^+, \]
\[ (pk(B,n(B,r2)))^-] \]

**Intruder capabilities**

\[ [\text{nil} | (M_1;M_2)^-,M_1^+] \]
\[ [\text{nil} | (M_1;M_2)^-,M_2^+] \]
\[ [\text{nil} | M^-,M_2^-, (M_1;M_2)^+] \]
\[ [\text{nil} | M^-, (sk(i,M))^+] \]
\[ [\text{nil} | M^-, (pk(Ke,M))^+] \]

**Equational Theory - Algebraic properties**

\[ B = \{ (X ; Y) ; Z = X ; (Y ; Z) \} \]
\[ \Delta = \{ \text{pk}(Ke,sk(Ke,X)) = X, \]
\[ \text{sk}(Ke,pk(Ke,X)) = X \} \]
Examples of execution

Needham-Schroeder Public Key: Attack State Pattern

:: r2 ::

[\textit{nil}, (pk(B, A; N_A))^-, (pk(A, N_A; n(B, r2)))^+, (pk(B, n(B, r2)))^- | \textit{nil} ]

& SS & \{ n(B, r2) \in I, IK \}
Examples of execution

Needham-Schroeder Public Key: Search State Space

Diagram showing the state space search process in a tree structure.
Examples of execution

**Needham-Schroeder Public Key: Initial State**

\[
\begin{align*}
[\text{nil} & \mid (pk(i, n(b, r2)))^-, (n(b, r2))^+, \text{nil}] \ \& \\
[\text{nil} & \mid (pk(i, a; n(a, r1)))^-, (a; n(a, r1))^+, \text{nil}] \ \& \\
[\text{nil} & \mid (n(b, r2))^-, (pk(b, n(b, r2)))^+, \text{nil}] \ \& \\
[\text{nil} & \mid (a; n(a, r1))^-, (pk(b, a; n(a, r1)))^+, \text{nil}] \ \& \\
:: \ r1 :: \\
[\text{nil} & \mid (pk(i, a; n(a, r1)))^+, (pk(a, n(a, r1); n(b, r2)))^-, (pk(i, n(b, r2)))^+, \text{nil}] \ \& \\
:: \ r2 :: \\
[\text{nil} & \mid (pk(b, a; n(a, r1)))^-, (pk(a, n(a, r1); n(b, r2)))^+, (pk(b, n(b, r2)))^-, \text{nil}] \\
\end{align*}
\]
Examples of execution

Needham-Schroeder Public Key: Attack sequence

1. \((pk(i, a; n(a, r1)))^+\)
2. \((pk(i, n(b, r2)))^-\)
3. \((a; n(a, r1))^+\)
4. \((a; n(a, r1))^-\)
5. \((pk(b, a; n(a, r1)))^+\)
6. \((pk(b, a; n(a, r1)))^-\)
7. \((pk(a, n(a, r1); n(b, r2)))^+\)
8. \((pk(a, n(a, r1); n(b, r2)))^-\)
9. \((pk(i, n(b, r2)))^+\)
10. \((pk(i, n(b, r2)))^-\)
11. \((n(b, r2))^+\)
12. \((n(b, r2))^-\)
13. \((pk(b, n(b, r2)))^+\)
14. \((pk(b, n(b, r2)))^-\)
Example: Needham-Schroeder-Lowe Protocol

Protocol (text-book)
\[ A \rightarrow B : pk(B, A; N_A) \]
\[ B \rightarrow A : pk(A, N_A; N_B; B) \]
\[ A \rightarrow B : pk(B, N_B) \]

Protocol (strand spaces)
\[ :: r1 :: [\text{nil} \mid (pk(B, A; n(A, r1)))^+, \]
\[ (pk(A, n(A, r1); N_B; B))^-, \]
\[ pk(B, N_B)^+] \]
\[ :: r2 :: [\text{nil} \mid (pk(B, A; N_A))^-, \]
\[ (pk(A, N_A; n(B, r2); B))^+, \]
\[ (pk(B, n(B, r2)))^-] \]

Intruder capabilities
\[ [\text{nil} \mid (M_1; M_2)^-, M_1^+] \]
\[ [\text{nil} \mid (M_1; M_2)^-, M_2^+] \]
\[ [\text{nil} \mid M_1^-, M_2^-, (M_1; M_2)^+] \]
\[ [\text{nil} \mid M^-, (sk(i, M))^+] \]
\[ [\text{nil} \mid M^-, (pk(Ke, M))^+] \]

Equational Theory - Algebraic properties
\[ B = \{ (X ; Y) ; Z = X ; (Y ; Z) \} \]
\[ \Delta = \{ pk(Ke, sk(Ke, X)) = X, \]
\[ sk(Ke, pk(Ke, X)) = X \} \]
Needham-Schroeder-Lowe: Attack State Pattern

:: r2 ::

\[ [\text{nil}, (pk(B, A; N_A))^-], (pk(A, N_A; n(B, r2); B))^+, (pk(B, n(B, r2)))^- | \text{nil} ] \]

& SS & \{ n(B, r2) \in I, IK \}
Needham-Schroeder-Lowe: Search State Space
Example: NSL-xor Protocol

Protocol (text-book)

\[ A \rightarrow B : pk(B, A; N_A) \]
\[ B \rightarrow A : pk(A, N_A; N_B \oplus B) \]
\[ A \rightarrow B : pk(B, N_B) \]

Protocol (strand spaces)

:: \( r_1 :: [\text{nil} | (pk(B, A; n(A, r_1)))^+, (pk(A, n(A, r_1); N_B \oplus B))^-, pk(B, N_B)^+] \)

:: \( r_2 :: [\text{nil} | (pk(B, A; N_A))^-, (pk(A, N_A; n(B, r_2) \oplus B))^+, (pk(B, n(B, r_2)))^-] \)

Intruder capabilities

\[ [\text{nil} | (M_1; M_2)^-, M_1^+] \]
\[ [\text{nil} | (M_1; M_2)^-, M_2^+] \]
\[ [\text{nil} | M_1^-, M_2^-, (M_1; M_2)^+] \]
\[ [\text{nil} | NS_1^-, NS_2^-, (NS_1 \oplus NS_2)^+] \]
\[ [\text{nil} | \text{null}^+] \]
\[ [\text{nil} | M^-, (sk(i, M))^+] \]
\[ [\text{nil} | M^-, (pk(Ke, M))^+] \]

Equational Theory - Algebraic properties

\[ B = \{ (X \oplus Y) \oplus Z = X \oplus (Y \oplus Z), X \oplus Y = Y \oplus X \} \]
\[ \Delta = \{ pk(Ke, sk(Ke, X)) = X, sk(Ke, pk(Ke, X)) = X, NS \oplus NS = \text{null}, NS1 \oplus NS1 \oplus NS2 = NS2, NS \oplus \text{null} = NS \} \]
NSL-xor: Attack State Pattern

:: r2 ::

\[ [\text{nil}, (pk(B, A; NS_A))^- , (pk(A, NS_A; n(B, r2) \oplus B))^+, \\
(\text{pk}(B, n(B, r2)))^- | \text{nil} ] \]

& \text{SS} & \{ n(B, r2) \in \mathcal{I}, \text{IK} \}
Examples of execution

NSL-xor: Search State Space
Examples of execution

**NSL-xor: Initial State**

\[
\begin{align*}
&\text{[nil | (pk(i, a; n(a, r1)))}, (a; n(a, r1))}, \text{nil] &
\text{[nil | (pk(i, b \oplus i \oplus n(b, r1)))}, (b \oplus i \oplus n(b, r1))}, \text{nil] &
\text{[nil | (a; n(a, r1))}, (pk(b, a; n(a, r1)))}, \text{nil] &
\text{[nil | (n(b, r2))}, (pk(b, n(b, r2)))}, \text{nil] &
\text{[nil | (b \oplus i)}}, (b \oplus i \oplus n(b, r2))}, \text{nil] &
\end{align*}
\]

:: r1 ::

\[
\begin{align*}
&\text{[nil | (pk(i, a; n(a, r1)))}, (pk(a, n(a, r1); n(b, r2) \oplus b))}, \text{nil] &
\end{align*}
\]

:: r2 ::

\[
\begin{align*}
&\text{[nil | (pk(b, a; n(a, r1)))}, (pk(a, n(a, r1); n(b, r2) \oplus b))}, \text{nil] &
\end{align*}
\]
NSL-xor: Attack sequence

1. $(pk(i, a; n(a, r1)))^+$
2. $(pk(i, n(b, r2)))^-$
3. $(a; n(a, r1))^+$
4. $(a; n(a, r1))^-$
5. $(pk(b, a; n(a, r1)))^+$
6. generatedByIntruder$(b \oplus i)$
7. $(pk(b, a; n(a, r1)))^-$
8. $(pk(a, n(a, r1); n(b, r2); b))^+$
9. $(pk(a, n(a, r1); n(b, r2); b))^-$
10. $(pk(i, n(b, r2) \oplus b \oplus i))^+$
11. $(pk(i, n(b, r2) \oplus b \oplus i))^-$
12. $(n(b, r2) \oplus b \oplus i)^+$
13. $(b \oplus i)^-$
14. $(n(b, r2) \oplus b \oplus i)^+$
15. $(n(b, r2))^+$
16. $(n(b, r2))^-$
17. $(pk(b, n(b, r2)))^+$
18. $(pk(b, n(b, r2)))^-$
Example: NSL-homomorphism Protocol

Protocol (text-book)

\[ A \rightarrow B : pk(B, A; N_A) \]
\[ B \rightarrow A : pk(A, N_A; N_B; B) \]
\[ A \rightarrow B : pk(B, N_B) \]

Protocol (strand spaces)

\[ :: r1 :: [nil \mid (pk(B, A; n(A, r1)))^+, (pk(A, n(A, r1); N_B; B))^-, pk(B, N_B)^+] \]
\[ :: r2 :: [nil \mid (pk(B, A; N_A))^-, (pk(A, N_A; n(B, r2); B))^+, (pk(B, n(B, r2)))^-] \]

Intruder capabilities

\[ :: [nil \mid (M_1; M_2)^-, M^+_1] \]
\[ :: [nil \mid (M_1; M_2)^-, M^+_2] \]
\[ :: [nil \mid M^+_1, M^-_2, (M_1; M_2)^+] \]
\[ :: [nil \mid M^-, (pk(Ke, M))^+] \]
\[ :: [nil \mid (pk(i, M))^+, M^+] \]

Equational Theory - Algebraic properties

\[ B = \{ (X ; Y) ; Z = X ; (Y ; Z) \} \]
\[ \Delta = \{ pk(Ke, X; Y) = pk(Ke, X); pk(Ke, Y) \} \]
NSL-homomorphism: Attack State Pattern

:: r2 ::

\[
[\text{nil}, (pk(B, A; N_A))^-, (pk(A, N_A; n(B, r2); B))^+, (pk(B, n(B, r2)))^- | \text{nil}] \\
& SS & \{n(B, r2) \in \mathcal{I}, IK\}
\]
NSL-homomorphism: Search State Space
NSL-homomorphic: Initial State

\[
\begin{align*}
\text{Nil} &\mid (pk(a,i)^-, (pk(a,n(b,r2)))^-, (pk(i,a); pk(a,n(b,r2)))^+, \text{Nil}) \\
\text{Nil} &\mid (pk(i,n(b,r2)))^-, (n(b,r2))^+, \text{Nil} \\
\text{Nil} &\mid (n(b,r2))^-, (pk(b,n(b,r2)))^+, \text{Nil} \\
\text{Nil} &\mid (pk(a,NI); pk(a,n(b,r2)); pk(a,b))^-, (pk(a,n(b,r2)); pk(a,b))^+, \text{Nil} \\
\text{Nil} &\mid (pk(a,n(b,r2)); pk(a,b))^-, (pk(a,n(b,r2)))^+, \text{Nil} \\
\text{Nil} &\mid (pk(i,n(b,r2)); pk(i,n(a,r1)); pk(i,a))^-, (pk(i,n(b,r2)))^+, \text{Nil} \\
:: &\ r1 :: \\
\text{Nil} &\mid (pk(a,i;n(b,r2))^-, (pk(i,n(b,r2); n(a,r1); a))^+, \text{Nil} \\
:: &\ r2 :: \\
\text{Nil} &\mid (pk(b,a;NI))^-, (pk(a,NI;n(b,r2); b))^+, (pk(b,n(b,r2)))^-, \text{Nil}
\end{align*}
\]
NSL-homomorphism: Attack sequence

1. $\text{generatedByIntruder}(pk(a, i))$
2. $\text{generatedByIntruder}(pk(b, a; NI))$
3. $(pk(b, a; NI))^{-}$
4. $(pk(a, NI; n(b, r2); b))^{+}$
5. $(pk(a, NI); pk(a, n(b, r2)); pk(a, b))^{-}$
6. $(pk(a, n(b, r2)); pk(a, b))^{+}$
7. $(pk(a, n(b, r2)); pk(a, b))^{-}$
8. $(pk(a, n(b, r2)))^{+}$
9. $(pk(a, i))^{-}$
10. $(pk(a, n(b, r2)))^{-}$
11. $(pk(i, a); pk(a, n(b, r2)))^{+}$
12. $(pk(a, i; n(b, r2)))^{-}$
13. $(pk(i, n(b, r1); n(a, r1); a))^{+}$
14. $(pk(i, n(b, r2)); pk(i, n(a, r1)); pk(i, a))^{-}$
15. $(pk(i, n(b, r2)))^{+}$
16. $(pk(i, n(b, r2)))^{-}$
17. $(n(b, r2))^{+}$
18. $(n(b, r2))^{-}$
19. $(pk(b, n(b, r2)))^{+}$
20. $(pk(b, n(b, r2)))^{-}$
Example: Diffie-Hellman Protocol

Protocol (text-book)
\[ A \rightarrow B : (A; B; \exp(g, N_A)) \]
\[ B \rightarrow A : (A; B; \exp(g, N_B)) \]
\[ A \rightarrow B : e(\exp(\exp(g, N_B), N_A), \sec(A, B)) \]

Protocol (strand spaces) Intruder capabilities
\[
\begin{align*}
\text{:: } r_1, r_2 & \:: [\text{nil} | (A; B; \exp(g, n(A, r1)))^+, \\
& (A; B; XB)^-, \\
& e(\exp(XB, n(A, r1)), \sec(A, r2))^+] \\
\text{:: } r_3 & \:: [\text{nil} | (A; B;XA)^-, \\
& (A; B; \exp(g, n(B, r3)))^+, \\
& (e(\exp(XA, n(B, r3)), S))^+] 
\end{align*}
\]

Equational Theory Algebraic properties
\[ B = \{ (X \ast Y) \ast Z = X \ast (Y \ast Z), (X \ast Y) = Y \ast X \} \]
\[ \Delta = \{ \text{dec}(K, \text{enc}(K, X)) = X, \exp(\exp(W, Y), Z) = \exp(W, Y \ast Z) \} \]
Diffie-Hellman: Attack State Pattern

:: r' :: [(A; B; Y)^-, (B; A; exp(g, n(B, r')))^+, (e(exp(Y, n(B, r')), sec(a, r'')))^- | nil] 
\& SS \& (sec(a, r'') \in \mathcal{I}, \ IK)
Diffie-Hellman: Initial State

\[
\begin{align*}
[\text{nil} & \mid \text{exp}(g, n(a, r))^-, Z^-, \text{exp}(g, Z \ast n(a, r))^+] \land \\
[\text{nil} & \mid \text{exp}(g, Z \ast n(a, r))^-, e(\text{exp}(g, Z \ast n(a, r)), \text{sec}(a, r''))^-, \text{sec}(a, r'')^+] \land \\
[\text{nil} & \mid \text{exp}(g, n(b, r'))^-, W^-, \text{exp}(g, W \ast n(b, r'))^+] \land \\
[\text{nil} & \mid \text{exp}(g, W \ast n(b, r'))^-, \text{sec}(a, r'')^-, e(\text{exp}(g, W \ast n(b, r'))), \text{sec}(a, r'')^+] \land \\
[\text{nil} & \mid (a; b; \text{exp}(g, n(b, r')))^-, (b; \text{exp}(g, n(b, r')))^+] \land \\
[\text{nil} & \mid (b; \text{exp}(g, n(b, r')))^-, \text{exp}(g, n(b, r'))^+] \land \\
[\text{nil} & \mid (a; B'; \text{exp}(g, n(a, r)))^-, (B'; \text{exp}(g, n(a, r)))^+] \land \\
[\text{nil} & \mid (B'; \text{exp}(g, n(a, r)))^-, \text{exp}(g, n(a, r))^+] \land \\
:: r' :: \\
[\text{nil} & \mid (a; b; \text{exp}(g, W))^-, (a; b; \text{exp}(g, n(b, r')))^+, e(\text{exp}(g, W \ast n(b, r'))), \text{sec}(a, r'')^-] \land \\
:: r'', r :: \\
[\text{nil} & \mid (a; B'; \text{exp}(g, n(a, r)))^+, (a; B'; \text{exp}(g, Z))^-, e(\text{exp}(g, Z \ast n(a, r)), \text{sec}(a, r''))^+] 
\end{align*}
\]
Diffie-Hellman: Attack sequence

1. $(a; b; \exp(g, W))^{-}$
2. $(a; b; \exp(g, n(b, r')))^{+}$
3. $(a; b; \exp(g, n(b, r')))^{-}$
4. $(b; \exp(g, n(b, r')))^{+}$
5. $(b; \exp(g, n(b, r')))^{-}$
6. $(\exp(g, n(b, r')))^{+}$
7. $(\exp(g, n(b, r')))^{-}$
8. $W^{-}$
9. $\exp(g, W * n(b, r'))^{+}$
10. $(a; B'; \exp(g, n(a, r)))^{+}$
11. $(a; B'; \exp(g, n(a, r)))^{-}$
12. $(B'; \exp(g, n(a, r)))^{+}$
13. $(B'; \exp(g, n(a, r)))^{-}$
14. $(\exp(g, n(a, r)))^{+}$
15. $(\exp(g, n(a, r)))^{-}$
16. $Z^{-}$
17. $\exp(g, Z * n(a, r))^{+}$
18. $(a; B'; \exp(g, Z))^{-}$
19. $e(\exp(g, Z * n(a, r)), \sec(a, r''))^{+}$
20. $e(\exp(g, Z * n(a, r)), \sec(a, r''))^{-}$
21. $\exp(g, Z * n(a, r))^{-}$
22. $\sec(a, r'')^{+}$
23. $\exp(g, W * n(b, r'))^{-}$
24. $\sec(a, r'')^{-}$
25. $e(\exp(g, W * n(b, r')), \sec(a, r''))^{+}$
26. $e(\exp(g, W * n(b, r')), \sec(a, r''))^{-}$
Many thanks