Relating Multiset Rewriting and Process Algebra for Immediate Decryption Protocols

Iliano Cervesato

iliano@itd.nrl.navy.mil

ITT Industries, inc @ NRL Washington, DC

http://www.cs.stanford.edu/~iliano

Joint work with S. Bistarelli, G. Lenzini, and F. Martinelli
Objective

- Relate specification languages for security protocols
  - MSR $\leftrightarrow$ strands [CSFW’00]
  - MSR $\leftrightarrow$ linear logic [MFPS’00]
  - MSR $\leftrightarrow$ Process Algebras

Non-Objective (for now)
- Reachability analysis $\leftrightarrow$ bisimulation
  - Verification methodologies not considered
Why MSR?

- Model of specification underlies numerous languages and tools
  - CIL/CAPSL
  - NRL Protocol Analyzer
  - Paulson’s Isabelle specifications
  - Murϕ
  - ...

- Simple and well-understood foundations
  - Distributed systems
    - Petri nets
  - Linear logic
  - Rewriting theory
Multiset Rewriting + Existentials

- msets of 1\textsuperscript{st}-order atomic formulas
- Rules:
  \[ r: F(x) \rightarrow \exists n. G(x, n) \]
- Application
  \[ M_1 \xrightarrow{r} M_2 \]
  \[ M', F(t) \xrightarrow{r} M', G(t, c) \]
  \( c \) not in \( M_1 \)

- This is MSR 1.0

MSR 2.0:
+ strong typing
+ constraints
+ domain-specific enhancements
Which Process Algebra?

“PA”

- Inspired to
  - CCS
  - $\pi$-calculus
- Only primitives used for protocols
- As a programming language for protocols
  - Reachability
  - Not simulation/equivalence
“PA”

- **Sequential processes**
  \[ P ::= 0 \mid a(t).P \mid \_a(t).P \mid \nu x.P \]

- **Parallel processes**
  \[ Q ::= 0 \mid P \parallel Q \mid !P \parallel Q \]

- **(P, ||, 0) monoid**
  - Equivalence \( \equiv \)

- **Reaction**
  \[ t = [\theta]t' \]
  \[ Q \parallel a(t).P \parallel a(t').P' \rightarrow Q \parallel P \parallel [\theta]P' \]
MSR $\Leftrightarrow$ PA … in General

- Very different paradigms
  - MSR
    - state transition
  - PA
    - contact evolution
- Non trivial
  - MSR $\rightarrow$ PA: granularity of actions
  - PA $\rightarrow$ MSR: excise state
  - Reachability-preserving
  - Non bijective
- Many attempts in the literature
  - Chemical abstract machine, ...
MSR $\Leftrightarrow$ PA … for Protocols

Much simpler!

- Take natural specifications
  - in MSR
  - in PA

- Bijective correspondence
  - (to a large extent)
MSR for Security Protocols

- Fixed predicates
  - $N(m)$: Network messages
  - $I(m)$: Intruder info.
  - $A_i(t_1, ..., t_{ni})$: Role states
  - $Pr, PrvK, PubK, ...$: Persistent info.

- Fixed format
  - Protocol given as set of roles
  - Dolev-Yao intruder spec.

- (more freedom in MSR 2.0)
Roles in MSR

• One instantiation rule
  \[ \pi(x) \rightarrow \exists n. A_0(x,n), \pi(x) \]

• Several execution rules
   Send
  \[ A_i(z) \rightarrow A_{i+1}(z), N(t) \]
   Receive
  \[ A_i(z), N(t) \rightarrow A_{i+1}(z,x_t) \]

Captures only immediate decryption protocols
NSPK (initiator) in MSR

\[ \pi_A(A,B) \rightarrow A_0(A,B), \pi_A(A,B) \]

\[ A_0(A,B) \rightarrow \exists N_A. A_1(A,B,N_A), N(\{N_A,A\}_{KB}) \]

\[ A_1(A,B, N_A), N(\{N_A,N_B\}_{KA}) \rightarrow A_2(A,B,N_A,N_B) \]

\[ A_2(A,B,N_A,N_B) \rightarrow A_3(A,B,N_A,N_B), N(\{N_B\}_{KB}) \]

where \[ \pi_A(A,B) = Pr(A), PrvK(A,K_A^{-1}), \]
\[ Pr(B), PubK(B,K_B) \]
MSR Configurations

- **Rules**
  - $U_\rho$  
    - Protocol roles
  - $\rho_I$  
    - Intruder role

- **State**
  - $N(t)$  
    - Network messages
  - $A_i(t)$  
    - Role state predicates
  - $\pi(t)$  
    - Persistent knowledge
  - $I(t)$  
    - Intruder knowledge
Security Protocols in PA

- Fixed set of names
  - $N_i, N_o, \pi, \mathcal{I}$

- Fixed structure of “Security Process”
  - $Q!\text{net} = ! N_i(x) \cdot N_o(x) \cdot 0$
  - Network process
  - Roles
  - $Q!\rho = \parallel_{\rho} P_{\rho}$
    - $! \pi(x). \forall n. P'$
    - Input on $N_o$
    - Output on $N_i$

- Dolev-Yao Intruder
  - Persistent information
  - Initial intruder knowledge

- Captures only immediate decryption protocols
NSPK (initiator) in PA

\[ \pi_A(A,B) \land \nu N_A \]

\[ N_i(\{N_A,A\}_{KB}) . \]

\[ N_o(\{N_A,N_B\}_{KA}) . \]

\[ N_i(\{N_B\}_{KB}) . \]

0
Process State

- $Q_i$  Replicated process
- $Q$  Unreplicated part
  - $Q_I$  Intruder knowledge
  - $Q_{\text{net}}$  Buffered network messages
  - $Q_\rho$  Roles in mid-execution
MSR into PA

- **Rules**
  - $U_\rho \rightarrow Q!_p + Q!_{net}$
    - Instantiation rule
    - “$A_i(z) \rightarrow A_{i+1}(z), N(t)$”
    - “$A_i(z), N(t) \rightarrow A_{i+1}(z,x_t)$”
  - $\rho_I \rightarrow Q!I$

- **State**
  - $N(t) \rightarrow Q_{net}$
  - $A_i(t) \rightarrow Q_\rho$
  - $\pi(t) \rightarrow Q!_\pi$
  - $I(t) \rightarrow Q_I$

Captures only immediate decryption protocols

MSR $\leftrightarrow$ PA

NSPK_{MSR} $\rightarrow$ NSPK_{PA}
PA into MSR

Essentially the inverse transformation

- $Q_{\rho} \rightarrow U_{\rho}$
  - Invent $A_i$'s
  - Carry over substitutions
- $Q_{\Pi} \rightarrow \rho_{\Pi}$

$\text{NSPK}_{PA} \rightarrow \text{NSPK}_{MSR}$
(for $\alpha$-convertible $A_i$'s)
The Intruder

1-1 correspondence, but ...

- \( I(\langle x_1, x_2 \rangle) \rightarrow I(x_1), I(x_2) \)
- \( I(x) \rightarrow I(x), I(x) \)
- \( I(x_1), I(x_2) \rightarrow I(\langle x_1, x_2 \rangle) \)
- \( I(\langle x_1, x_2 \rangle). I(x_1). 0 \)
- \( I(\langle x_1, x_2 \rangle). I(x_2). 0 \)
- \( I(x). I(x). I(x). 0 \)
- \( I(x_1). I(x_2). I(\langle x_1, x_2 \rangle). 0 \)
Correspondence

- Proof technique: weak bi-simulation
  - Observables
    - Network messages
    - Intruder knowledge
Delayed Decryption Protocols

- Arguments of $A_i$'s may be terms
- Explicit pattern matching in PA

• Add non-trivial complications
  - Requires proper scheduling of matchings
  - Matching after input may cause deadlock

• Solutions
  - WITS’03 unsatisfactory
  - Intermediate MSR with explicit scheduling
Conclusions

• Formal relation between MSR and PA
  - As used for security protocols
  - Non trivial (yet mostly bijective)
  - Technique similar to MSR <-> strands

... And future work
  - MSR 3.0
  - Strict comparison with spi-calculus
  - Relating methodologies