Relating Multiset Rewriting and Process Algebra for Security Protocol Specification

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 <-> strands [CSFW'00]
  - MSR <-> linear logic [MFPS'00]
  - MSR <-> Process Algebras

Non-Objective (for now)

- Reachability analysis <-> 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 1st-order atomic formulas
- Rules:

$$r: F(\mathbf{x}) \rightarrow \exists \mathbf{n}.\ G(\mathbf{x},\mathbf{n})$$

- Application

$$\underbrace{M_1}_{M',\ F(\mathbf{t})} \xrightarrow{r} \underbrace{M_2}_{M',\ G(\mathbf{t},\mathbf{c})}$$

$\mathbf{c}$ 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
  - π-calculus
- Only primitives used for protocols
- As a programming language for protocols
  - Reachability
  - Not simulation/equivalence

“PA”

- Sequential processes

  $P ::= 0 \mid a(x).P \mid \underline{a}(t).P \mid [x=t]P \mid \nu x.P$

- Parallel processes

  $Q ::= 0 \mid P \,||\, Q \mid !P \,||\, Q$

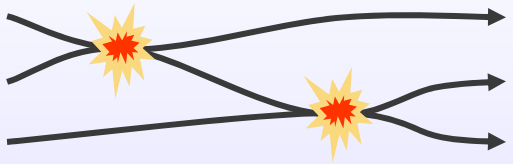
- (P, ||, 0) monoid
  - Equivalence ≡
- Reaction

$$Q \,||\, \underline{a}(t).P \,||\, a(x).P' \;\rightarrow\; Q \,||\, P \,||\, [t/x]P'$$

MSR ⇔ PA ... in General

- Very different paradigms
  - MSR
    - state transition

  - PA
    - contact evolution



- Non trivial
    - MSR -> PA: *granularity of actions*
    - PA -> MSR: *excise state*
  - Reachability-preserving
  - Non bijective

MSR ⇔ 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.
  - $A_i(t_1,\ldots,t_{ni})$ — 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

  $\pi(\mathbf{x}) \rightarrow \exists \mathbf{n}.\ A_0(\mathbf{x},\mathbf{n}),\ \pi(\mathbf{x})$

- Several execution rules
  - Send

    $A_i(\mathbf{z}) \rightarrow A_{i+1}(\mathbf{z}),\ N(t)$
  - Receive

    $A_i(t),\ N(t') \rightarrow A_{i+1}(\mathbf{z}_t,\mathbf{x}_{t'})$

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
  - ρ_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 name
  - N_i, N_o, π, I
- Fixed structure of "Security Process"

$Q_!$ {
- $Q_{!net} = !\, N_i(x).\, \underline{N_o}(x).\, 0$ — Network process
- $Q_{!\rho} = \|_\rho P_\rho$ — Roles
  - $!\, \pi(\mathbf{x}).\, \nu\mathbf{n}.\, P'$
    - input on N_o
    - output on $\underline{N_i}$
    - pattern matching
- $Q_{!I}$ — Dolev-Yao Intruder
- $Q_{!\pi}$ — Persistent information
- Q_{I0} — Initial intruder knowledge
}

NSPK (initiator) in PA

$\pi_A(A,B).\ \nu N_A$

$\underline{N_i}(\{N_A,A\}_{KB})\ .$

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

$\underline{N_i}(\{N_B\}_{KB})\ .$

0

Process State

- $Q_!$ Replicated process
- Q Unreplicated part
  - Q_I Intruder knowledge
  - Q_{net} Buffered network messages
  - Q_ρ Roles in mid-execution

MSR into PA

- Rules
  - $U\rho \rightarrow Q_{!\rho} + Q_{!net}$
    - Instantiation rule → "! $\pi(x)$. νn." prefix
    - "$A_i(z) \rightarrow A_{i+1}(z), N(t)$" → $\underline{N_i}(t).\ \langle r_{i+1} \rangle$
    - "$A_i(t), N(t') \rightarrow A_{i+1}(z_t, x_{t'})$" → $N_o(x).\ [x=t']\ [z=t]\ \langle r_{i+1} \rangle$
  - $\rho_I \rightarrow Q_{!I}$
- State
  - $N(t) \rightarrow Q_{net}$
  - $A_i(t) \rightarrow Q_\rho$
  - $\pi(t) \rightarrow Q_{!\pi}$
  - $I(t) \rightarrow Q_I$

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

(once reducing variable renamings)

$[x=x']$

PA into MSR

Essentially the inverse transformation

- $Q_{!\rho}$ → $U\rho$
  - Invent A_i's
  - Carry over substitutions
- $Q_{!I}$ → ρ_I

$NSPK_{PA} \rightarrow NSPK_{MSR}$
(for α-convertible A_i's)

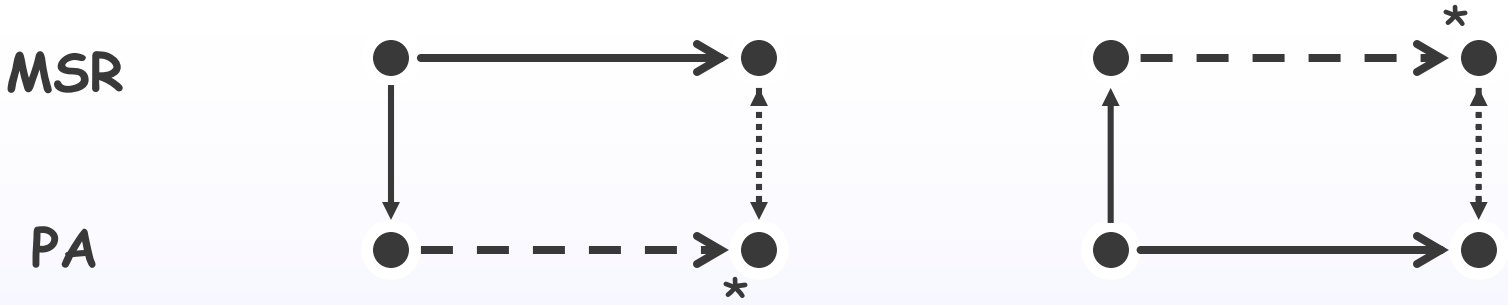
The Intruder

1-1 correspondence, but …

- $I(\langle x_1,x_2\rangle) \rightarrow I(x_1), I(x_2)$
- $I(x) \rightarrow I(x), I(x)$
- $I(x_1), I(x_2) \rightarrow I(\langle x_1,x_2\rangle)$

- $I(\langle x_1,x_2\rangle).\ \underline{I}(x_1).\ 0$
  $I(\langle x_1,x_2\rangle).\ \underline{I}(x_2).\ 0$
- $I(x).\ \underline{I}(x).\ \underline{I}(x).\ 0$
- $I(x_1).\ I(x_2).\ \underline{I}(\langle x_1,x_2\rangle).\ 0$

Correspondence



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

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