

Relating Multiset Rewriting and Process Algebra for Immediate Decryption Protocols



Fabio Martinelli `fabio.martinelli@iit.cnr.it`

CNR - Pisa, Italy

<http://www.iit.cnr.it/staff/fabio.martinelli/>

Joint work with S. Bistarelli, I. Cervesato, and G. Lenzini



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
 - Mur ϕ
 - ...
- Simple and well-understood foundations
 - Distributed systems
 - Petri nets
 - Linear logic
 - Rewriting theory



Multiset Rewriting + Existentials

- msets of 1st-order atomic formulas
- Rules:

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

- Application

$$\begin{array}{ccc} M_1 & \xrightarrow{r} & M_2 \\ \underbrace{} & & \underbrace{} \\ M', F(t) & \xrightarrow{r} & M', G(t,c) \\ & & c \text{ not in } M_1 \end{array}$$

- 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

$$P ::= 0 \mid a(t).P \mid \underline{a}(t).P \mid vx.P$$

- Parallel processes

$$Q ::= 0 \mid P \parallel Q \mid !P \parallel Q$$

- $(P, \parallel, 0)$ monoid

➤ Equivalence $\equiv$

- Reaction

$$t = [\theta]t'$$

$$Q \parallel \underline{a}(t).P \parallel a(t').P' \rightarrow Q \parallel P \parallel [\theta]P'$$

MSR $\Leftrightarrow$ PA ... in General

- Very different paradigms

- MSR

- state transition

- PA

- contact evolution

- Non trivial

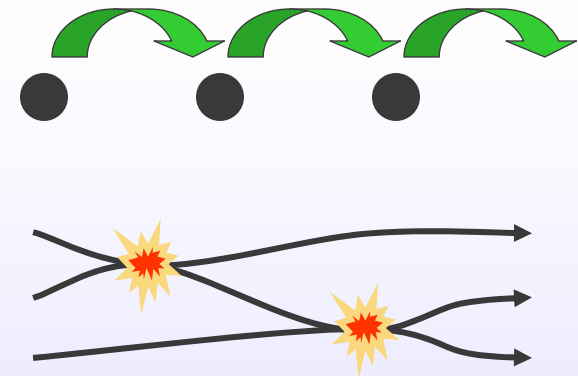
- MSR $\rightarrow$ PA: *granularity of actions*
 - PA $\rightarrow$ MSR: *excise state*

- 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)

What Protocols?

- Immediate decryption protocols
 - Received messages always decryptable

$A \rightarrow B : k$
 $B \rightarrow A : \dots$
 $A \rightarrow B : \{n\}_k$
 $B \rightarrow A : n$



~~$A \rightarrow B : \{n\}_k$
 $B \rightarrow A : \dots$
 $A \rightarrow B : k$
 $B \rightarrow A : n$~~

- Generic authentication protocols
 - Much more complicated
 - Forthcoming publication



MSR for Security Protocols

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

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

➤ Receive

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

Captures
only
immediate
decryption
protocols



NSPK (initiator) in MSR



$$\pi_A(A, B) \rightarrow A_0(A, B), \pi_A(A, B)$$

$$A_0(A, B) \rightarrow \exists N_A. A_1(A, B, N_A), N(\{N_A, A\}_{K_B})$$

$$A_1(A, B, N_A), N(\{N_A, N_B\}_{K_A}) \rightarrow A_2(A, B, N_A, N_B)$$

$$A_2(A, B, N_A, N_B) \rightarrow A_3(A, B, N_A, N_B), N(\{N_B\}_{K_B})$$

where $\pi_A(A, B) = Pr(A), PrvK(A, K_A^{-1}),$
 $Pr(B), PubK(B, K_B)$

MSR Configurations

- Rules

- U_{ρ}

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

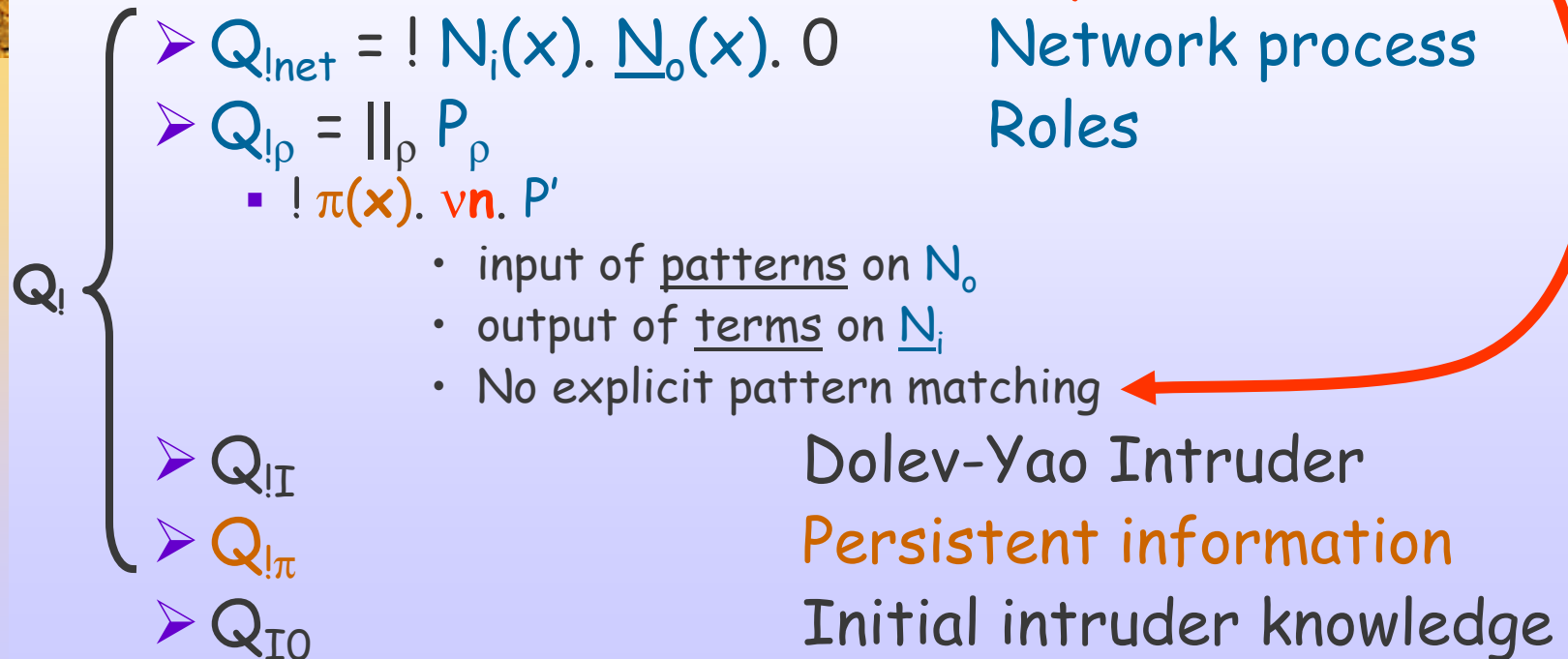


Security Protocols in PA

- Fixed set of name

➤ N_i, N_o, π, I

- Fixed structure of "Security Process"



Captures
only
immediate
decryption
protocols

NSPK (initiator) in PA

$\pi_A(A, B). \textcolor{red}{v}N_A$

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

$N_o(\{N_A, \textcolor{violet}{N}_B\}_{KA}).$

$\underline{N}_i(\{N_B\}_{KB}) .$

0



Process State

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



MSR into PA

Rules

$$\triangleright U_p \rightarrow Q_{!p} + Q_{!net}$$

▪ Instantiation rule

▪ " $A_i(z) \rightarrow A_{i+1}(z), N(t)$ "

▪ " $A_i(z), N(t) \rightarrow A_{i+1}(z, x_t)$ "

$\rightarrow$ " $! \pi(x). vn.$ " prefix

$\rightarrow \underline{N}_i(t). \langle r_{i+1} \rangle$

$\rightarrow N_o(t). \langle r_{i+1} \rangle$

$$\triangleright \rho_I \rightarrow Q_{!I}$$

State

$$\triangleright N(t) \rightarrow Q_{net}$$

$$\triangleright A_i(t) \rightarrow Q_p$$

$$\triangleright \pi(t) \rightarrow Q_{! \pi}$$

$$\triangleright I(t) \rightarrow Q_I$$

Captures
only
immediate
decryption
protocols

$$\text{NSPK}_{\text{MSR}} \rightarrow \text{NSPK}_{\text{PA}}$$

PA into MSR

Essentially the inverse transformation

- $Q_{!p} \rightarrow U_p$
 - Invent A_i 's
 - Carry over substitutions
- $Q_{!I} \rightarrow \rho_I$

$$\text{NSPK}_{PA} \rightarrow \text{NSPK}_{MSR}$$

(for α -convertible A_i 's)

The Intruder

1-1 correspondence, but ...

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

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

- $I(x) \rightarrow I(x), I(x)$

- $I(x). \underline{I}(x). \underline{I}(x). 0$

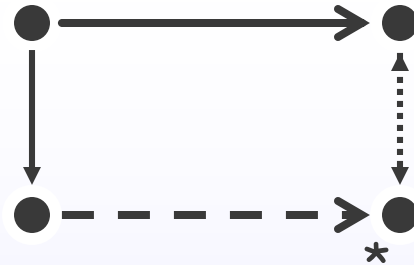
- $I(x_1), I(x_2) \rightarrow I(\langle x_1, x_2 \rangle)$

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

Correspondence

MSR

PA



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

Generic 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
 - Intermediate MSR with explicit scheduling
 - Forthcoming paper

Conclusions ...

- Formal relation between MSR and PA
 - As used for security protocols
 - Non trivial (yet mostly bijective)
 - Technique similar to MSR $\leftrightarrow$ strands

... and Future Work

- MSR 3.0
- Strict comparison with spi-calculus
- Relating methodologies

