ArmoredSoftware Semantics 0.0

ArmoredSoftware Crew
Information and Telecommunication Technology Center
The University of Kansas
palexand@ku.edu

November 24, 2014

Contents

List of Figures

Abstract

This document describes evolving ArmoredSoftware semantic definitions.

1 Introduction

2 SPI Calculus

Examples motivated by **?**.

2.1 Wide Mouth Frog

2.2 Needham Schroeder

$$
\begin{gathered}
A \rightarrow B : \{A^+, N_A\}_{B+} \text{ on } c \\
B \rightarrow A : \{N_A, N_B\}_{A+} \text{ on } c \\
A \rightarrow B : \{N_B\}_{B+} \text{ on } c
\end{gathered}
$$

$$
\begin{array}{rcl}
A & \triangleq & \overline{c}\langle\{(A, N_A)\}_{B+}\rangle. \\
 & & c(M). \\
 & & \mathsf{case}\ \{M\}_{A-}\ \mathsf{of}\ (N_A, nb)\ \mathsf{in} \\
 & & \overline{c}\langle\{nb\}_{B+}\rangle. \\
 & & A \\
B & \triangleq & c(M). \\
 & & \mathsf{case}\ \{M\}_{B-}\ \mathsf{of}\ (x, n)\ \mathsf{in} \\
 & & \overline{c}\langle\{(n, N_B)\}_{x+}\rangle. \\
 & & c(M). \\
 & & \mathsf{case}\ \{M\}_{B-}\ \mathsf{of}\ N_B\ \mathsf{in}\ B \\
sys & \triangleq & (\nu c)A \mid B
\end{array}
$$

$$\text{React Inter}\ \frac{}{\overline{m}\langle N\rangle.p \mid m(x).Q \rightarrow P \mid [x \rightarrow N]Q}$$

$$\text{Red Replace}\ \frac{}{!P > P \,|!P}$$

$$\text{Red Match}\ \frac{}{[M\ \mathsf{is}\ M]P > P}$$

$$\text{Red Let}\ \frac{}{\mathsf{let}\ (x, y) = (M, N)\ \mathsf{in}\ P > [x \rightarrow M][y \rightarrow N]P}$$

Note that we may want a more general let that matches more than pairs here. We'll see what the other inference rules give us.

$$\text{Red Zero}\ \frac{}{\mathsf{case}\ 0\ \mathsf{of}\ \ 0 : P\ suc(x) : Q > P}$$

$$\text{Red Suc}\ \frac{}{\mathsf{case}\ suc(M)\ \mathsf{of}\ \ 0 : P\ suc(x) : Q > [x \rightarrow M]Q}$$

I find the case rules over naturals quite crude.

$$\text{Red Sym Decrypt}\ \frac{}{\mathsf{case}\ \{M\}_k\ \mathsf{of}\ \{x\}_k\ \mathsf{in}\ P > [x \rightarrow M]P}$$

Additional proposed semantic rules for public/private key encryption and signature checking

$$\text{Red Asym Decrypt}\ \frac{}{\mathsf{case}\ \{M\}_{k^+}\ \mathsf{of}\ \{x\}_{k^-}\ \mathsf{in}\ P > [x \rightarrow M]P}$$

$$\text{Red Sig Check}\ \frac{}{\mathsf{case}\ \{|M|\}_{k^-}\ \mathsf{of}\ \{|x|\}_{k^+}\ \mathsf{in}\ P > [x \rightarrow M]P}$$

Do we really want a signature check that fails to get stuck? The M is available, but the signature check does not pass. This would work if we treat $\{|M|\}_{k^-}$ as only the signature block and not the message. A signed message may be best represented as a pair $(M, \{|M|\}_{k^-})$ allowing the message to be explicitly available. What that construct looks like is up in the air at this point.

This rule has a more serious problem as it allows us to reproduce a message from it's signature. Specifically, if we have $\{|M|\}_{k^-}$ and signature match is successful, then x is bound to M. That can't happen. Possibly the rule should look like this:

$$\text{Red Sig Check}\ \frac{}{\mathsf{case}\ \{|M|\}_{k^-}\ \mathsf{of}\ \{|x|\}_{k^+}\ \mathsf{in}\ P > [x \rightarrow |M]]P}$$

where $|M|$ is the hash and not the message itself. Maybe a signature check should look something like this:

$$\mathsf{let}\ (m, s) = (M, \{|M|\}_{k^-})\ \mathsf{in}\ \mathsf{case}\ s\ \mathsf{of}\ \{|M|\}_{k^+}\ \mathsf{in}\ P$$

A quick reduction gives:

$$[m \rightarrow M][s \rightarrow \{|M|\}_{k^-}]\mathsf{case}\ s\ \mathsf{of}\ \{|M|\}_{k^+}\ \mathsf{in}\ P$$

Substitution gives:

$$\mathsf{case}\ \{|M|\}_{k^-}\ \mathsf{of}\ \{|M|\}_{k^+}\ \mathsf{in}\ [m \rightarrow M][s \rightarrow \{|M|\}_{k^-}]P$$

Finally, using the signature reduction rule:

$$[m \rightarrow M][s \rightarrow \{|M|\}_{k^-}]P$$

If the signature does not match, the process hangs. Assume M$\neq N$:

$$\mathsf{let}\ (m, s) = (M, \{|N|\}_{k^-})\ \mathsf{in}\ \mathsf{case}\ s\ \mathsf{of}\ \{|M|\}_{k^+}\ \mathsf{in}\ P$$
$$[m \rightarrow M][s \rightarrow \{|N|\}_{k^-}]\mathsf{case}\ s\ \mathsf{of}\ \{|M|\}_{k^+}\ \mathsf{in}\ P$$
$$\mathsf{case}\ \{|N|\}_{k^-}\ \mathsf{of}\ \{|M|\}_{k^+}\ \mathsf{in}\ [m \rightarrow M][s \rightarrow \{|M|\}_{k^-}]P$$

This is pretty much what we want I think other than the signature check hanging on failure. I think that's what it should be, but signature check failure still results in a message that could be processed. I suppose the way to do it would be put the signature check process in parallel with a process that does not check it.

$$\text{Struct Nil}\,\frac{}{P \mid \mathbf{0} \equiv P}$$

$$\text{Struct Comm}\,\frac{}{P \mid Q \equiv Q \mid P}$$

$$\text{Struct Assoc}\,\frac{}{P \mid (Q \mid R) \equiv (P \mid Q) \mid R}$$

$$\text{Struct Switch}\,\frac{}{(\nu m)(\nu n)P \equiv (\nu n)(\nu m)P}$$

$$\text{Struct Drop}\,\frac{}{(\nu n)\mathbf{0} \equiv \mathbf{0}}$$

$$\text{Struct Extrusion}\,\frac{n \notin fv(P)}{(\nu n)(P \mid Q) \equiv P \mid (\nu n)Q}$$

$$\text{Struct Red}\,\frac{P > Q}{P \equiv Q}$$

$$\text{Struct Refl}\,\frac{}{P \equiv P}$$

$$\text{Struct Symm}\,\frac{P \equiv Q}{Q \equiv P}$$

$$\text{Struct Trans}\,\frac{P \equiv Q \qquad Q \equiv R}{P \equiv R}$$

$$\text{Struct Par}\,\frac{P \equiv P'}{P \mid Q \equiv P' \mid Q}$$

$$\text{Struct Res}\,\frac{P \equiv P'}{(\nu n)P \equiv (\nu n)P'}$$

$$\text{React Struct}\,\frac{P \equiv P' \qquad P' \to Q' \qquad Q' \equiv Q}{P \to Q}$$

$$\text{React Par}\,\frac{P' \to P'}{P \mid Q \to P' \mid Q}$$

$$\text{React Res}\,\frac{P' \to P'}{(\nu n)P \to (\nu n)P}$$

$$A \quad \triangleq \quad \overline{c}\langle\{(A, N\_A)\}_{B^+}\rangle.$$

2.3 Privacy CA Protocol

A Glossary

- $\mathbf{0}$ - null process
- $|M|$ - hash of M
- K^+ - public half of asymmetric key K
- K^- - private half of asymmetric key K
- $\{M\}_K$ - encrypt M with symmetric key K
- $\{M\}_{K^+}$ - encrypt M with the public key from K
- $\{M\}_{K^-}$ - decrypt M with the public key from K
- $\{|M|\}_{K^-}$ - sign M with the private key from K
- $\{|M|\}_{K^+}$ - check signature on M with the public key from K

- $(\nu x)P$ - new variable x defined in scope of P
- $\overline{c}\langle M\rangle$ - send M on channel c
- $c(M)$ - receive M on channel c
- $!P$ - infinite replication of P
- $P\ +\ Q$ - P or Q
- $P \mid Q$ - P in parallel with Q
- case $\{M\}_k$ of x in P - attempt to decrypt $\{M\}_k$ and bind to x in P if successful. Stuck if unsuccessful
- case $\{M\}_{k^-}$ of x in P - attempt to decrypt $\{M\}_{k^+}$ and bind to x in P if successful. Stuck if unsuccessful
- case $\{|M|\}_{k^+}$ of x in P - attempt to check signature $\{|M|\}_{k^-}$ and bind to x in P if successful. Stuck if unsuccessful
- case x of y $0 : P\ suc(x) : Q$ - case splitting over integers. x is bound in Q.
- let $(x, y) = M$ in y - match M to (x, y) binding x and y to pair elements in M
- $A \triangleq B$ - define an equivalence
- $A \rightarrow B : M$ on c - A sends B message M on channel c

$$\begin{aligned} A \ \triangleq\ & (\nu c)\ \overline{c}\langle M\rangle.\mathbf{0} \mid \\ & c(M).A \end{aligned}$$

