ArmoredSoftware Semantics 0.0

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

December 5, 2014

Contents

1	Intr	roduction	2	
2	SPI	Calculus	2	
	2.1	Wide Mouth Frog	2	
	2.2	Needham Schroeder	2	
	2.3	Privacy CA Protocol	5	
3 \$	Strand Spaces		5	
	3.1	Needham Schroder	5	
	3.2	Wide Mouth Frog	5	
A	Glo	ssary	5	

List of Figures

Abstract

This document describes evolving $\ensuremath{\mathsf{ARMOREDSOFTWARE}}$ semantic definitions.

1 Introduction

2 SPI Calculus

Examples motivated by ?.

2.1 Wide Mouth Frog

2.2 Needham Schroeder

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

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

$$A \rightarrow B: \{N_B\}_{B^+} \text{ on } c$$

$$\begin{split} A & \stackrel{\triangle}{=} \overline{c} \langle \{(A,N_A)\}_{B^+} \rangle. \\ & c(M). \\ & \operatorname{case} \ \{M\}_{A^-} \ \operatorname{of} \ (N_A,nb) \ \operatorname{in} \\ & \overline{c} \langle \{nb\}_{B^+} \rangle. \\ & A \\ B & \stackrel{\triangle}{=} c(M). \\ & \operatorname{case} \ \{M\}_{B^-} \ \operatorname{of} \ (x,n) \ \operatorname{in} \\ & \overline{c} \langle \{(n,N_B)\}_{x^+} \rangle. \\ & c(M). \\ & \operatorname{case} \ \{M\}_{B^-} \ \operatorname{of} \ N_B \ \operatorname{in} \ B \\ & sys & \stackrel{\triangle}{=} (\nu c)A \mid B \end{split}$$

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

Red Replace $\overline{P>P\mid P}$
Red Match $\overline{M\mid P>P}$

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

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.

Red Zero case 0 of
$$0: P suc(x): Q > P$$

I find the case rules over naturals quite crude.

Red Sym Decrypt case
$$\{M\}_k$$
 of $\{x\}_k$ in $P > [x \to M]P$

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

Red Asym Decrypt 1 case
$$\{M\}_{k^+}$$
 of $\{x\}_{k^-}$ in $P>[x\to M]P$
Red Asym Decrypt 2 case $\{M\}_{k^-}$ of $\{x\}_{k^+}$ in $P>[x\to M]P$

The previous two rules capture the essence of asymmetric key pairs. Specifically, encrypt with one and decrypt with the other.

Assume |M| is the hash and not the message itself and $\{|M|\}_{k^-}$ is the hash encrypted with private key, k^- . Thus, a signed message is the pair $(M, \{|M|\}_{k^-})$ consisting of the message and the signed hash. Given this, a successful signature check looks something like this:

$$\begin{split} & | \text{let } (m,s) = (M,\{|M|\}_{k^-}) \text{ in case } s \text{ of } \{x\}_{k^+} \text{ in } [\![x \text{ is } |m|]\!] P \\ & > [m \to M][s \to \{|M|\}_{k^-}] \text{case } s \text{ of } \{x\}_{k^+} \text{ in } [\![x \text{ is } |m|]\!] P \\ & > \text{case } \{|M|\}_{k^-} \text{ of } \{x\}_{k^+} \text{ in } [m \to M][s \to \{|M|\}_{k^-}] [\![x \text{ is } |m|]\!] P \\ & > [\![x \to |M]\!] [m \to M][s \to \{|M|\}_{k^-}] [x \text{ is } |m]\!] P \\ & > [\![M|\text{ is } |M|]\!] [x \to |M|] [m \to M][s \to \{|M|\}_{k^-}] P \\ & > [x \to |M|] [m \to M][s \to \{|M|\}_{k^-}] P \end{split}$$

This is precisely what we want. Specifically, P with m replaced by the message M and s replaced by the decrypted signature, |M|, produced by the signature check. It is unlikely that |M| will be used in P, but it is available.

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

$$\begin{split} & | \text{let } (m,s) = (M,\{|N|\}_{k^-}) \text{ in case } s \text{ of } \{x\}_{k^+} \text{ in } [\![x \text{ is } |m|]\!] P \\ & > [m \to M][s \to \{|N|\}_{k^-}] \text{case } s \text{ of } \{x\}_{k^+} \text{ in } [\![x \text{ is } |m|]\!] P \\ & > \text{case } \{|N|\}_{k^-} \text{ of } \{x\}_{k^+} \text{ in } [m \to M][s \to \{|N|\}_{k^-}] [\![x \text{ is } |m|]\!] P \\ & > [\![x \to |N|]\!] [m \to M][s \to \{|M|\}_{k^-}] [\![x \text{ is } |m|]\!] P \\ & > [\![N|\text{ is } |M|]\!] [x \to |N|] [m \to M][s \to \{|M|\}_{k^-}] P \end{split}$$

The process is stuck when |N| is |M| fails because $N \neq M$.

Now assume the wrong private key was used to sign the message hash. Specifically, $j \neq k$:

$$\begin{split} & | \text{let } (m,s) = (M,\{|M|\}_{j^-}) \text{ in case } s \text{ of } \{x\}_{k^+} \text{ in } [\![x \text{ is } |m|]\!] P \\ & > [m \to M][s \to \{|M|\}_{j^-}] \text{case } s \text{ of } \{x\}_{k^+} \text{ in } [\![x \text{ is } |m|]\!] P \\ & > \text{case } \{|M|\}_{j^-} \text{ of } \{x\}_{k^+} \text{ in } [m \to M][s \to \{|M|\}_{k^-}] [\![x \text{ is } |m|]\!] P \end{split}$$

The process is stuck when $\{|M|\}_{j^-}$ does not unify with $\{x\}_{k^+}$.

Do we really want a signature check that fails to get stuck? I think so. M is available, but the signature check is stuck. A signed message is best represented as a pair $(M, \{|M|\}_{k^-})$ allowing the message to be explicitly available.

Struct Nil
$$P \mid \mathbf{0} \equiv P$$

Struct Comm $P \mid Q \equiv Q \mid P$

Struct Assoc $P \mid (Q \mid R) \equiv (P \mid Q) \mid R$

Struct Switch $(\nu m)(\nu n)P \equiv (\nu n)(\nu m)P$

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

Struct Extrusion $n \notin fv(P)$

Struct Red $P > Q$
 $P \equiv Q$

Struct Refl $P \equiv P$

Struct Symm $P \equiv Q$

Struct Symm $P \equiv Q$

Struct Trans $P \equiv P$

Struct Par $P \equiv P$

Struct Par $P \equiv P$

Struct Par $P \equiv P$

Struct Res $P \equiv P$

React Struct Res $P \equiv P$

React Struct $P \equiv P$

Struct Res $P \equiv P$

React Struct $P \equiv P$

Struct Res $P \equiv P$

React Struct $P \equiv P$

React Struct $P \equiv P$

React $P \Rightarrow P$

React $P \Rightarrow P$

React $P \Rightarrow P$

React $P \Rightarrow P$

$$A \quad \stackrel{\Delta}{=} \quad \overline{c} \langle \{ (A, N_{-}A) \}_{B^{+}} \rangle.$$

2.3 Privacy CA Protocol

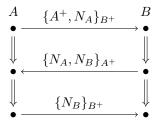
3 Strand Spaces

3.1 Needham Schroder

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

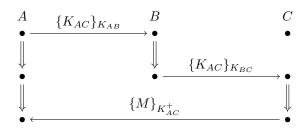
$$B \rightarrow A: \{N_A, N_B\}_{A^+} \text{ on } c$$

$$A \rightarrow B: \{N_B\}_{B^+} \text{ on } c$$



3.2 Wide Mouth Frog

$$A \rightarrow B: \{K_{AC}\}_{K_{AB}} \text{ on } c$$
 $B \rightarrow C: \{K_{AC}\}_{K_{BC}} \text{ on } c$ $A \rightarrow C: \{M\}_{K_{AC}} \text{ on } c$



A Glossary

 \bullet 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
- \bullet !P infinite replication of P
- $\bullet P + Q P \text{ or } Q$
- $P \mid Q$ 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
- $A \stackrel{\Delta}{=} B$ define an equivalence
- $A \rightarrow B : M$ on c A sends B message M on channel c

$$A \stackrel{\Delta}{=} (\nu c) \, \overline{c} \langle M \rangle. \mathbf{0} \mid c(M). A$$