

# Protocol Composition for Arbitrary Primitives <sup>1</sup> (t.b.p. at CSF 2010)

Ștefan Ciobâcă    Véronique Cortier

LSV, ENS Cachan & CNRS & INRIA

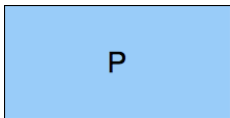
LORIA, CNRS & INRIA

April 15th, 2010

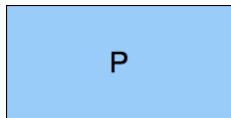
---

<sup>1</sup>This work has been partly supported by the ANR SeSur AVOTÉ.

# Protocol composition

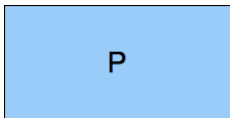


# Protocol composition

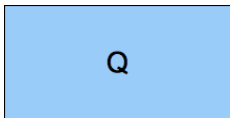


$\models \phi$

# Protocol composition

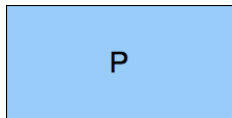


$\models \phi$

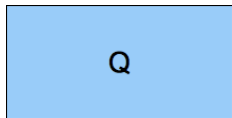


$\models \psi$

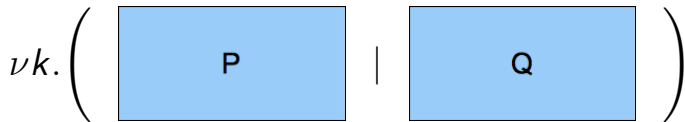
# Protocol composition



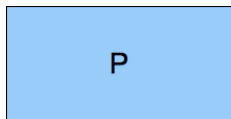
$\models \phi$



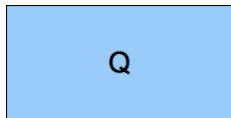
$\models \psi$



# Protocol composition



$\models \phi$



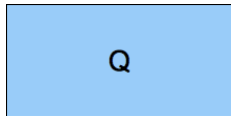
$\models \psi$

$\nu k. \left( \begin{array}{|c|} \hline P \\ \hline \end{array} \mid \begin{array}{|c|} \hline Q \\ \hline \end{array} \right) \models ?$

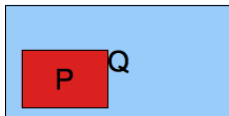
# Protocol composition



$\models \phi$



$\models \psi$

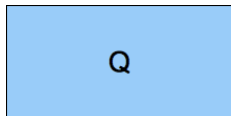


$\models ?$

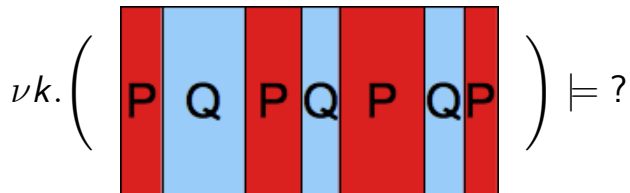
# Protocol composition



$\models \phi$

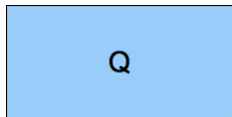


$\models \psi$



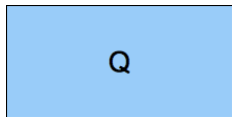
$\models ?$

# Protocol composition





*e.g. symmetric encryption  
signatures*



*e.g. asymmetric encryption  
another symmetric encryption*

$P, Q, \dots ::=$  processes  
 $\quad | \mathbf{0}$  the empty process

# Process syntax

$P, Q, \dots ::=$  processes

- | **0** the empty process
- |  $\nu x$
- | **in**( $x$ )
- |  $x := t$

# Process syntax

$P, Q, \dots ::=$  processes

- | **0** the empty process
- |  $\nu x$
- | **in**( $x$ )
- |  $x := t$
- | **out**( $t$ )

$P, Q, \dots$	$::=$	processes
		<b>0</b> the empty process
		$\nu x$
		<b>in</b> ( $x$ )
		$x := t$
		<b>out</b> ( $t$ )
		$[s = t]$ <i>modulo an equational theory</i>

# Process syntax

$P, Q, \dots$	$::=$	processes
		<b>0</b> the empty process
		$\nu x$
		<b>in</b> ( $x$ )
		$x := t$
		<b>out</b> ( $t$ )
		$[s = t]$ <i>modulo an equational theory</i>
		$P \cdot Q$

$P, Q, \dots$	$::=$	processes
		<b>0</b> the empty process
		$\nu x$
		<b>in</b> ( $x$ )
		$x := t$
		<b>out</b> ( $t$ )
		$[s = t]$ <i>modulo an equational theory</i>
		$P \cdot Q$
		$P \mid Q$
		$!P$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q = \nu z \cdot \mathbf{out}(y) \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{0}$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q = \nu z \cdot \mathbf{out}(y) \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q = \nu z \cdot \mathbf{out}(y) \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q \not\vdash z$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q = \nu z \cdot \mathbf{out}(y) \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q \not\equiv z$$

$$P \cdot Q$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q = \nu z \cdot \mathbf{out}(y) \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q \not\vdash z$$

$$P \cdot Q \vdash z$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q = \nu z \cdot \mathbf{out}(y) \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q \not\vdash z$$

$$P \cdot Q \vdash z$$

(even if the encryption function is not the same in  $P, Q$ )

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q' = \nu z \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{in}(z') \cdot \mathbf{out}(\mathit{dec}(z', y)) \cdot \mathbf{0}$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q' = \nu z \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{in}(z') \cdot \mathbf{out}(\mathit{dec}(z', y)) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q'$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q' = \nu z \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{in}(z') \cdot \mathbf{out}(\mathit{dec}(z', y)) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q' \not\sim z$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q' = \nu z \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{in}(z') \cdot \mathbf{out}(\mathit{dec}(z', y)) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q' \not\sim z$$

$$P \cdot Q'$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q' = \nu z \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{in}(z') \cdot \mathbf{out}(\mathit{dec}(z', y)) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q' \not\vdash z$$

$$P \cdot Q' \vdash z$$

$$P = \nu x \cdot \nu y \cdot \mathbf{out}(\{x\}_y)$$

$$Q' = \nu z \cdot \mathbf{out}(\{z\}_x) \cdot \mathbf{in}(z') \cdot \mathbf{out}(\mathit{dec}(z', y)) \cdot \mathbf{0}$$

$$\nu x' \cdot \nu y' \cdot (x := x') \cdot (y := y') \cdot Q' \not\vdash z$$

$$P \cdot Q' \vdash z$$

(because  $P, Q'$  use the same encryption/decryption algorithm)

$$R = \nu x \cdot (\textit{establish key } y) \cdot \nu z \cdot [x = y] \cdot \mathbf{out}(z) \cdot \mathbf{0}$$

$$R = \nu x \cdot (\text{establish key } y) \cdot \nu z \cdot [x = y] \cdot \mathbf{out}(z) \cdot \mathbf{0}$$

$$R' = \nu x \cdot \nu y \cdot \nu z \cdot [x = y] \cdot \mathbf{out}(z) \cdot \mathbf{0} \not\vdash z$$

$$R = \nu x \cdot (\textit{establish key } y) \cdot \nu z \cdot [x = y] \cdot \mathbf{out}(z) \cdot \mathbf{0}$$

$$R' = \nu x \cdot \nu y \cdot \nu z \cdot [x = y] \cdot \mathbf{out}(z) \cdot \mathbf{0} \not\vdash z$$

$$P = (y := x)$$

$$R = \nu x \cdot (\text{establish key } y) \cdot \nu z \cdot [x = y] \cdot \mathbf{out}(z) \cdot \mathbf{0}$$

$$R' = \nu x \cdot \nu y \cdot \nu z \cdot [x = y] \cdot \mathbf{out}(z) \cdot \mathbf{0} \not\vdash z$$

$$P = (y := x)$$

$$R'' = \nu x \cdot P \cdot \nu z \cdot [x = y] \cdot \mathbf{out}(z) \cdot \mathbf{0} \vdash z$$

## Definition

A linear process is a process without replication or parallel composition.

# Main theorem

## Definition

A linear process is a process without replication or parallel composition.

## Theorem (Simplified)

- 1 Let  $P$  be a linear process over  $\Sigma_a$  and  $E_a$ .
- 2 Let  $Q$  be a linear process over  $\Sigma_b$  and  $E_b$ .
- 3 Let  $R$  be a closed linear process obtained by arbitrarily interleaving  $P$  and  $Q$  (over  $\Sigma_a \cup \Sigma_b$  and  $E_a \cup E_b$ ).

Then  $R$  reveals a secret  $s$  in  $E_a \cup E_b$  if  $(\nu \text{fv}(P)) \cdot P$  or  $(\nu \text{fv}(Q)) \cdot Q$  reveal  $s$  or a shared key in  $E_a$  (resp.  $E_b$ )<sup>a</sup>.

---

<sup>a</sup>or if  $P$  or  $Q$  instantiate two shared keys to the same value

$$\begin{aligned} A \rightarrow B &: g^a, \text{mac}(g^a, k_0) \\ B \rightarrow A &: g^b, \text{mac}(g^b, k_0) \\ A &: k := (g^b)^a \\ B &: k := (g^a)^b \end{aligned}$$

$$\begin{aligned}
 A \rightarrow B &: g^a, \text{mac}(g^a, k_0) \\
 B \rightarrow A &: g^b, \text{mac}(g^b, k_0) \\
 A &: k := (g^b)^a \\
 B &: k := (g^a)^b
 \end{aligned}$$

Let

$$P_1 = \nu x \cdot \mathbf{out}(g(x)) \cdot \mathbf{out}(\text{mac}(g(x), x_k)) \cdot \mathbf{in}(z) \cdot \mathbf{in}(z') \cdot [z' = \text{mac}(z, x_k)] \cdot y_1 := f(x, z)$$

$$\begin{aligned}
 A \rightarrow B &: g^a, \text{mac}(g^a, k_0) \\
 B \rightarrow A &: g^b, \text{mac}(g^b, k_0) \\
 A &: k := (g^b)^a \\
 B &: k := (g^a)^b
 \end{aligned}$$

Let

$$P_1 = \nu x \cdot \mathbf{out}(g(x)) \cdot \mathbf{out}(\text{mac}(g(x), x_k)) \cdot \mathbf{in}(z) \cdot \mathbf{in}(z') \cdot [z' = \text{mac}(z, x_k)] \cdot y_1 := f(x, z)$$

$$\mathcal{F}_a = \{f/2, g/1, \text{mac}/2, n_1/0, \dots\}, E_a = \{f(x, g(y)) = f(y, g(x))\}.$$

Let

$$P_2 = \nu y \cdot \mathbf{out}(g(y)) \cdot \mathbf{out}(mac(g(y), x_k)) \cdot \mathbf{in}(z) \cdot \mathbf{in}(z') \cdot [z' = mac(z, x_k)] \cdot y_2 := f(y, z)$$

Then

$$P = \nu x_k \cdot (P_1 \mid P_2)$$

is a model for the entire Diffie-Hellman protocol.

# Application

Let  $Q_1 = \nu x_s \cdot \mathbf{out}(enc(x_s, y_1))$  and  $Q_2 = \mathbf{in}(x) \cdot y := dec(x, y_2)$ .

$$Q = \nu y_k \cdot (y_1 := y_k \cdot Q_1 \mid y_2 := y_k \cdot Q_2)$$

$Q$  preserves the secrecy of  $s$ .

Let  $Q_1 = \nu x_s \cdot \mathbf{out}(enc(x_s, y_1))$  and  $Q_2 = \mathbf{in}(x) \cdot y := dec(x, y_2)$ .

$$Q = \nu y_k \cdot (y_1 := y_k \cdot Q_1 \mid y_2 := y_k \cdot Q_2)$$

$Q$  preserves the secrecy of  $s$ .

The sequential composition of Diffie-Hellman and  $Q$ :

$$W = \nu x_k \cdot (P_1 \cdot Q_1 \mid P_2 \cdot Q_2)$$

preserves the secrecy of  $x_s$  by the composition theorem.

$$P = \nu \tilde{k} \cdot (P_1 \mid P_2) \not\vdash x_k, y_k$$

$$P = \nu \tilde{k} \cdot (P_1 \mid P_2) \not\vdash x_k, y_k$$

$$Q = \nu x \cdot (x_k := x \cdot Q_1 \mid y_k := x \cdot Q_2) \not\vdash x_k, y_k, x_s$$

$$P = \nu \tilde{k} \cdot (P_1 \mid P_2) \not\vdash x_k, y_k$$

$$Q = \nu x \cdot (x_k := x \cdot Q_1 \mid y_k := x \cdot Q_2) \not\vdash x_k, y_k, x_s$$

By the composition theorem:

$$W = \nu \tilde{k} \cdot (P_1 \cdot Q_1 \mid P_2 \cdot Q_2) \not\vdash x_s$$

$$P' = \nu \tilde{k} \cdot !(P_1 \mid P_2) \not\vdash x_k, y_k$$

$$P' = \nu \tilde{k} \cdot !(P_1 \mid P_2) \not\vdash x_k, y_k$$

$$Q' = !(\nu x \cdot (x_k := x \cdot Q_1 \mid y_k := x \cdot Q_2)) \not\vdash x_k, y_k, x_s$$

$$P' = \nu \tilde{k} \cdot !(P_1 \mid P_2) \not\vdash x_k, y_k$$

$$Q' = !(\nu x \cdot (x_k := x \cdot Q_1 \mid y_k := x \cdot Q_2)) \not\vdash x_k, y_k, x_s$$

By the composition theorem:

$$W' = \nu \tilde{k} \cdot !(P_1 \cdot Q_1 \mid P_2 \cdot Q_2) \not\vdash x_s$$

- 1 sometimes two protocols use the same cryptographic primitives (encryption) and therefore we cannot model them by using two disjoint equational theories.
- 2 tagging consists in adding a small “tag” to each encrypted/hashed message.
- 3  $untag_c(tag_c(x)) = x$

- 1 sometimes two protocols use the same cryptographic primitives (encryption) and therefore we cannot model them by using two disjoint equational theories.
- 2 tagging consists in adding a small “tag” to each encrypted/hashed message.
- 3  $untag_c(tag_c(x)) = x$

$$\begin{array}{ll} \mathbf{out}(enc(s, k)) & \rightarrow \mathbf{out}(enc(tag_a(s), k)) \\ \mathbf{in}(x) \cdot y := dec(x, k) & \rightarrow \mathbf{in}(x) \cdot [tag_a(untag_a(dec(x, k))) = dec(x, k)] \cdot \\ & y := untag_a(dec(x, k)) \end{array}$$

## Theorem (Simplified)

- 1 Let  $P^a$  be the  $a$ -tagged version of a linear process  $P$ .
- 2 Let  $Q^b$  be the  $b$ -tagged version of a linear process  $Q$ .
- 3 Let  $R^{ab}$  be a closed linear process obtained by arbitrarily interleaving  $P^a$  and  $Q^b$ .

Then  $R^{ab}$  reveals a secret  $s$  if  $P$  or  $Q$  reveal  $s$  or a shared key  $a$ .

---

<sup>a</sup>or if  $P$  or  $Q$  instantiate two shared keys to the same value

## Theorem (Simplified)

- 1 Let  $P^a$  be the  $a$ -tagged version of a linear process  $P$ .
- 2 Let  $Q^b$  be the  $b$ -tagged version of a linear process  $Q$ .
- 3 Let  $R^{ab}$  be a closed linear process obtained by arbitrarily interleaving  $P^a$  and  $Q^b$ .

Then  $R^{ab}$  reveals a secret  $s$  if  $P$  or  $Q$  reveal  $s$  or a shared key  $a$ .

---

<sup>a</sup>or if  $P$  or  $Q$  instantiate two shared keys to the same value

Proof technique:

$$\text{enc}(\text{tag}_a(u), v) \mapsto \text{enc}_a(u, v)$$

$$\text{enc}(\text{tag}_b(u), v) \mapsto \text{enc}_b(u, v)$$

- ① a composition result in an equational setting
- ② generic theorem of composition: transforms any attack on a composite trace into an attack on a trace of one of the two protocols

- 1 a composition result in an equational setting
- 2 generic theorem of composition: transforms any attack on a composite trace into an attack on a trace of one of the two protocols
- 3 preservation of more generic properties (authentication, generic temporal logic)
- 4 preservation of behavioural equivalences
- 5 negative tests
- 6 sharing “harmless” primitives (pair)