Composition of Password-based Protocols

Stéphanie Delaune[1], Steve Kremer[1] and Mark Ryan[2]

[1] LSV, ENS de Cachan, CNRS & INRIA, France

[2] School of Computer Science, University of Birmingham, UK

FormaCrypt
November 30, 2007

Formal protocol analysis and composition

Nowadays tools exist that succeed in automatically analysing complex protocols, e.g. AVISPA and ProVerif

But: protocols are analysed in isolation

Other protocols may be executed in parallel

Need for compositional security guarantees

Formal protocol analysis and composition

Nowadays tools exist that succeed in automatically analysing complex protocols, e.g. AVISPA and ProVerif

But: protocols are analysed in isolation

Other protocols may be executed in parallel

Need for compositional security guarantees

Cryptographic process calculi and composition

Cryptographic pi calculi, e.g., the applied pi calculus or the spi calculus are well-suited for reasoning about composition

if P_1 is secure and P_2 is secure then $P_1 \mid P_2$ is secure

There are two main reasons for this

1. processes are shown secure in the presence of an arbitrary environment
2. processes do not share any secrets (this is due to the scope operator)

One would like to show that

if $\nu s.P_1$ is secure and $\nu s.P_2$ is secure then $\nu s.(P_1 \mid P_2)$ is secure

which does not hold in general

Note that $\nu s.(P_1 \mid P_2)$ differs from $\nu s.P_1 \mid \nu s.P_2$

Cryptographic process calculi and composition

Cryptographic pi calculi, e.g., the applied pi calculus or the spi calculus are well-suited for reasoning about composition

$$\text{if } P_1 \approx S_1 \text{ and } P_2 \approx S_2 \text{ then } P_1 \mid P_2 \approx S_1 \mid S_2$$

There are two main reasons for this

1. processes are shown secure in the presence of an arbitrary environment
2. processes do not share any secrets (this is due to the scope operator)

One would like to show that

$$\text{if } \nu s.P_1 \text{ is secure and } \nu s.P_2 \text{ is secure then } \nu s.(P_1 \mid P_2) \text{ is secure}$$

which does not hold in general

Note that $\nu s.(P_1 \mid P_2)$ differs from $\nu s.P_1 \mid \nu s.P_2$

Cryptographic process calculi and composition

Cryptographic pi calculi, e.g., the applied pi calculus or the spi calculus are well-suited for reasoning about composition

$$\text{if } P_1 \approx S_1 \text{ and } P_2 \approx S_2 \text{ then } P_1 \mid P_2 \approx S_1 \mid S_2$$

There are two main reasons for this

1. processes are shown secure in the presence of an arbitrary environment
2. processes do not share any secrets (this is due to the scope operator)

One would like to show that

$$\text{if } \nu s.P_1 \text{ is secure and } \nu s.P_2 \text{ is secure then } \nu s.(P_1 \mid P_2) \text{ is secure}$$

which does not hold in general

Note that $\nu s.(P_1 \mid P_2)$ differs from $\nu s.P_1 \mid \nu s.P_2$

Cryptographic process calculi and composition

Cryptographic pi calculi, e.g., the applied pi calculus or the spi calculus are well-suited for reasoning about composition

$$\text{if } P_1 \approx S_1 \text{ and } P_2 \approx S_2 \text{ then } P_1 \mid P_2 \approx S_1 \mid S_2$$

There are two main reasons for this

1. processes are shown secure in the presence of an arbitrary environment
2. processes do not share any secrets (this is due to the scope operator)

One would like to show that

$$\text{if } \nu s.P_1 \text{ is secure and } \nu s.P_2 \text{ is secure then } \nu s.(P_1 \mid P_2) \text{ is secure}$$

which does not hold in general

Note that $\nu s.(P_1 \mid P_2)$ differs from $\nu s.P_1 \mid \nu s.P_2$

Cryptographic process calculi and composition

Cryptographic pi calculi, e.g., the applied pi calculus or the spi calculus are well-suited for reasoning about composition

$$\text{if } P_1 \approx S_1 \text{ and } P_2 \approx S_2 \text{ then } P_1 \mid P_2 \approx S_1 \mid S_2$$

There are two main reasons for this

1. processes are shown secure in the presence of an arbitrary environment
2. processes do not share any secrets (this is due to the scope operator)

One would like to show that

$$\text{if } \nu s.P_1 \text{ is secure and } \nu s.P_2 \text{ is secure then } \nu s.(P_1 \mid P_2) \text{ is secure}$$

which does not hold in general

Note that $\nu s.(P_1 \mid P_2)$ differs from $\nu s.P_1 \mid \nu s.P_2$

Cryptographic process calculi and composition

Cryptographic pi calculi, e.g., the applied pi calculus or the spi calculus are well-suited for reasoning about composition

$$\text{if } P_1 \approx S_1 \text{ and } P_2 \approx S_2 \text{ then } P_1 \mid P_2 \approx S_1 \mid S_2$$

There are two main reasons for this

1. processes are shown secure in the presence of an arbitrary environment
2. processes do not share any secrets (this is due to the scope operator)

One would like to show that

$$\text{if } \nu s.P_1 \text{ is secure and } \nu s.P_2 \text{ is secure then } \nu s.(P_1 \mid P_2) \text{ is secure}$$

which does not hold in general

Note that $\nu s.(P_1 \mid P_2)$ differs from $\nu s.P_1 \mid \nu s.P_2$

Guessing attacks

Solution: do not share secrets between protocols, but this is not always possible

Passwords: it is not realistic that users never re-use the same password

In this talk we investigate the question:

if $\nu p.P_1$ and $\nu p.P_2$ are resistant against guessing attacks on p
is $\nu p.(P_1 \mid P_2)$ also resistant against guessing attacks on p?

guessing or dictionnary attacks consists of two phases

1. the attacker one or several sessions of a protocol
2. the attacker tries offline each of the possible passwords (out of a dictionnary) on the data collected during the first phase

Guessing attacks

Solution: do not share secrets between protocols, but this is not always possible

Passwords: it is not realistic that users never re-use the same password

In this talk we investigate the question:

if $\nu p.P_1$ and $\nu p.P_2$ are resistant against guessing attacks on p
is $\nu p.(P_1 \mid P_2)$ also resistant against guessing attacks on p?

guessing or dictionnary attacks consists of two phases

1. the attacker one or several sessions of a protocol
2. the attacker tries offline each of the possible passwords (out of a dictionnary) on the data collected during the first phase

Guessing attacks

Solution: do not share secrets between protocols, but this is not always possible

Passwords: it is not realistic that users never re-use the same password

In this talk we investigate the question:

if $\nu p.P_1$ and $\nu p.P_2$ are resistant against guessing attacks on p
is $\nu p.(P_1 \mid P_2)$ also resistant against guessing attacks on p?

guessing or dictionnary attacks consists of two phases

1. the attacker one or several sessions of a protocol
2. the attacker tries offline each of the possible passwords (out of a dictionnary) on the data collected during the first phase

Guessing attacks

Solution: do not share secrets between protocols, but this is not always possible

Passwords: it is not realistic that users never re-use the same password

In this talk we investigate the question:

if $\nu p.P_1$ and $\nu p.P_2$ are resistant against guessing attacks on p
is $\nu p.(P_1 \mid P_2)$ also resistant against guessing attacks on p?

Passive guessing or dictionnary attacks consists of two phases

1. the attacker eavesdrops one or several sessions of a protocol
2. the attacker tries offline each of the possible passwords (out of a dictionnary) on the data collected during the first phase

Guessing attacks

Solution: do not share secrets between protocols, but this is not always possible

Passwords: it is not realistic that users never re-use the same password

In this talk we investigate the question:

if $\nu p.P_1$ and $\nu p.P_2$ are resistant against guessing attacks on p
is $\nu p.(P_1 \mid P_2)$ also resistant against guessing attacks on p?

Active guessing or dictionnary attacks consists of two phases

1. the attacker interacts with one or several sessions of a protocol
2. the attacker tries offline each of the possible passwords (out of a dictionnary) on the data collected during the first phase

Terms and equational theories

We consider a simple process language inspired by the applied pi calculus to describe protocols

Messages are modeled using terms

- Abstract algebra given by a signature, *i.e.* a set of function symbols with arities
- Equivalence relation ($=_E$) on terms induced by an equational theory

Example (equational theory)

Consider the signature $\Sigma_{\mathsf{enc}} = \{\mathsf{sdec}, \mathsf{senc}, \mathsf{adec}, \mathsf{aenc}, \mathsf{pk}, \langle\ \rangle, \mathsf{proj}_1, \mathsf{proj}_2\}$

$$\begin{array}{rclrcl} \mathsf{sdec}(\mathsf{senc}(x,y),y) & = & x & \mathsf{proj}_i(\langle x_1, x_2\rangle) & = & x_i \quad (i \in \{1,2\}) \\ \mathsf{senc}(\mathsf{sdec}(x,y),y) & = & x & \mathsf{adec}(\mathsf{aenc}(x,\mathsf{pk}(y)),y) & = & x \end{array}$$

Frames and deduction

Terms are regrouped into frames: a set of secrets + a substitution

$$\nu\tilde{n}.(\{^{M_1}/_{x_1}\} \mid \ldots \mid \{^{M_n}/_{x_n}\})$$

Definition (Deduction)

$\nu\tilde{n}.\sigma \vdash_{\mathsf{E}} M$ iff there exists N such that $fn(N) \cap \tilde{n} = \emptyset$ and $N\sigma =_{\mathsf{E}} M$.
We call N a *recipe* of the term M.

Example

Let $\phi = \nu k, s_1.\{^{\mathsf{senc}(\langle s_1, s_2\rangle, k)}/_{x_1}, {}^{k}/_{x_2}\}$

	Recipe
$\phi \vdash_{\mathsf{E}_{\mathsf{enc}}} k$	x_2
$\phi \vdash_{\mathsf{E}_{\mathsf{enc}}} s_1$	$\mathsf{proj}_1(\mathsf{sdec}(x_1, x_2))$
$\phi \vdash_{\mathsf{E}_{\mathsf{enc}}} s_2$	s_2

Static equivalence

Definition (Static equivalence)

M and N are equal in ϕ, written $(M =_{\mathsf{E}} N)\phi$, if $\phi =_\alpha \nu\tilde{n}.\sigma$, $M\sigma =_{\mathsf{E}} N\sigma$, and $\tilde{n} \cap (fn(M) \cup fn(N)) = \emptyset$.

ϕ_1 and ϕ_2 are statically equivalent, $\phi_1 \approx_{\mathsf{E}} \phi_2$, when:

- $dom(\phi_1) = dom(\phi_2)$, and
- for all terms M, N, $(M =_{\mathsf{E}} N)\phi_1$ iff $(M =_{\mathsf{E}} N)\phi_2$

Example

$$\phi = \nu k.\{^{\mathsf{senc}(s_0,k)}/_{x_1}, {}^{k}/_{x_2}\} \not\approx \nu k.\{^{\mathsf{senc}(s_1,k)}/_{x_1}, {}^{k}/_{x_2}\} = \phi'$$

because of the test $(\mathsf{sdec}(x_1, x_2), s_0)$
However,

$$\nu k.\{^{\mathsf{senc}(s_0,k)}/_{x_1}\} \approx \nu k.\{^{\mathsf{senc}(s_1,k)}/_{x_1}\}$$

Syntax of the process language

Plain processes

$$
\begin{array}{ll}
P, Q, R := & \\
\quad 0 & \text{null process} \\
\quad P \mid Q & \text{parallel composition} \\
\quad \mathrm{in}(x).P & \text{message input} \\
\quad \mathrm{out}(M).P & \text{message output} \\
\quad \text{if } M = N \text{ then } P \text{ else } Q & \text{conditional}
\end{array}
$$

Extended processes $A, B, C := P \mid A \mid B \mid \nu n.A \mid \{^M/_x\}$

Frame of a process : $\phi(A)$ obtained by replacing plain processes by 0

Example

$$
\begin{aligned}
A = \;& \nu s, k_1.(\mathrm{in}(x).\text{ if } \mathrm{sdec}(x, k_1) = s \text{ then } \mathrm{out}(a) \mid \{^{\mathrm{senc}(s,k_1)}/_x\} \mid \\
& \qquad \nu k_2.\mathrm{out}(\mathrm{senc}(s, k_2))) \\
\phi(A) = \;& \nu s, k_1.(0 \mid \{^{\mathrm{senc}(s,k_1)}/_x\} \mid \nu k_2.0)
\end{aligned}
$$

Structural equivalence: the smallest equivalence relation closed by application of evaluation contexts and such that

Par-0	$A \mid 0$	$\equiv$	A	New-Par	$A \mid \nu n.B$	$\equiv$	$\nu n.(A \mid B)$
Par-C	$A \mid B$	$\equiv$	$B \mid A$				$n \notin fn(A)$
Par-A	$(A \mid B) \mid C$	$\equiv$	$A \mid (B \mid C)$	New-C	$\nu n_1.\nu n_2.A$	$\equiv$	$\nu n_2.\nu n_1.A$

Operational semantics: smallest relation between extended processes which is closed under structural equivalence ($\equiv$) and such that

In	$\mathrm{in}(x).P \xrightarrow{in(M)} P\{^M/_x\}$	
Out	$\mathrm{out}(M).P \xrightarrow{out(M)} P \mid \{^M/_x\}$	where x is a fresh variable
Then	if $M = N$ then P else $Q \xrightarrow{\tau} P$	where $M =_{\mathsf{E}} N$
Else	if $M = N$ then P else $Q \xrightarrow{\tau} Q$	where $M \neq_{\mathsf{E}} N$
Cont.	$\dfrac{A \xrightarrow{\ell} B}{C[A] \xrightarrow{\ell} C[B]}$	where C is an evaluation context if $\ell = in(M)$ then $\phi(C[A]) \vdash_{\mathsf{E}} M$

Structural equivalence: the smallest equivalence relation closed by application of evaluation contexts and such that

Par-0	$A \mid 0$	$\equiv$	A	New-Par	$A \mid \nu n.B$	$\equiv$	$\nu n.(A \mid B)$
Par-C	$A \mid B$	$\equiv$	$B \mid A$				$n \notin fn(A)$
Par-A	$(A \mid B) \mid C$	$\equiv$	$A \mid (B \mid C)$	New-C	$\nu n_1.\nu n_2.A$	$\equiv$	$\nu n_2.\nu n_1.A$

Operational semantics: smallest relation between extended processes which is closed under structural equivalence ($\equiv$) and such that

In	$\mathrm{in}(x).P \xrightarrow{in(M)} P\{^M/_x\}$	
Out	$\mathrm{out}(M).P \xrightarrow{out(M)} P \mid \{^M/_x\}$	where x is a fresh variable
Then	$\text{if } M = N \text{ then } P \text{ else } Q \xrightarrow{\tau} P$	where $M =_{\mathsf{E}} N$
Else	$\text{if } M = N \text{ then } P \text{ else } Q \xrightarrow{\tau} Q$	where $M \neq_{\mathsf{E}} N$
Cont.	$\dfrac{A \xrightarrow{\ell} B}{C[A] \xrightarrow{\ell} C[B]}$	where C is an evaluation context if $\ell = in(M)$ then $\phi(C[A]) \vdash_{\mathsf{E}} M$

Definition of guessing attacks

Definition from [Baudet05] (inspired from [Corin et al.03])

Definition (Passive guessing attacks)

$\nu w.\phi$ is resistant to guessing attacks against w iff

$$\nu w.(\phi \mid \{^{w}/_{x}\}) \approx \nu w.(\phi \mid \nu w'.\{^{w'}/_{x}\})$$

Definition (Active guessing attacks)

A is resistant to guessing attack against w if, for every process B such that $A \rightarrow^* B$, we have that $\phi(B)$ is resistant to guessing attacks against w.

Definition of guessing attacks

Definition from [Baudet05] (inspired from [Corin et al.03])

Definition (Passive guessing attacks)

$\nu w.\phi$ is resistant to guessing attacks against w iff

$$\nu w.(\phi \mid \{^{w}/_{x}\}) \approx \nu w.(\phi \mid \nu w'.\{^{w'}/_{x}\})$$

Definition (Active guessing attacks)

A is resistant to guessing attack against w if, for every process B such that $A \rightarrow^* B$, we have that $\phi(B)$ is resistant to guessing attacks against w.

Composing resistance against passive guessing attacks

Proposition

The three following statements are equivalent:

1. $\nu w.\phi \mid \{^{w}/_{x}\} \approx \nu w.\phi \mid \nu w'.\{^{w'}/_{x}\}$ [Baudet05]
2. $\phi \approx \nu w.\phi$ [Corin et al.03]
3. $\phi \approx \phi\{^{w'}/_{w}\}$

It follows from the last point that passive guessing attacks do compose!

Corollary

If $\nu w.\phi_1$ and $\nu w.\phi_2$ are resistant to guessing attacks against w then $\nu w.(\phi_1 \mid \phi_2)$ is also resistant to guessing attacks against w.

Composing resistance against passive guessing attacks

Proposition

The three following statements are equivalent:

1. $\nu w.\phi \mid \{^{w}/_{x}\} \approx \nu w.\phi \mid \nu w'.\{^{w'}/_{x}\}$ [Baudet05]
2. $\phi \approx \nu w.\phi$ [Corin et al.03]
3. $\phi \approx \phi\{^{w'}/_{w}\}$

It follows from the last point that passive guessing attacks do compose!

Corollary

If $\nu w.\phi_1$ and $\nu w.\phi_2$ are resistant to guessing attacks against w then $\nu w.(\phi_1 \mid \phi_2)$ is also resistant to guessing attacks against w.

Unbounded number of sessions for free

A consequence for password-only protocols:
if one session of the protocol is safe against a passive adversary then an unbounded number of sessions are safe against a passive adversary

Example (EKE protocol [BellovinMerritt92])

A → B :	$\mathrm{senc}(\mathrm{pk}(k)), w)$	(EKE.1)
B → A	$\mathrm{senc}(\mathrm{aenc}(r, \mathrm{pk}(k)), w)$	(EKE.2)
A → B	$\mathrm{senc}(na, r)$	(EKE.3)
B → A	$\mathrm{senc}(\langle na, nb\rangle, r)$	(EKE.4)
A → B	$\mathrm{senc}(nb, r)$	(EKE.5)

$$\phi = \nu k, r, na, nb. \quad \{{}^{\mathrm{senc}(\mathrm{pk}(k),w)}/_{x_1}, {}^{\mathrm{senc}(\mathrm{aenc}(r,\mathrm{pk}(k)),w)}/_{x_2}, {}^{\mathrm{senc}(na,r)}/_{x_3}, {}^{\mathrm{senc}(\langle na,nb\rangle,r)}/_{x_4}, {}^{\mathrm{senc}(nb,r)}/_{x_5}\}$$

We indeed have that $\nu w.(\phi \mid \{{}^{w}/_{x}\}) \approx \nu w, w'.(\phi \mid \{{}^{w'}/_{x}\})$

Resistance against active guessing attacks does not compose

Active resistance against guessing attacks does not compose in general!

Counter-example

Suppose the following equational theory

$$\mathsf{f}_3(\mathsf{f}_2(\mathsf{f}_1(x,y),x,z),y) = x$$

$$\begin{array}{rcl} A_1 & = & \nu n_1.\mathsf{out}(\mathsf{f}_1(w,n_1)).\mathsf{in}(x).\mathsf{out}(\mathsf{f}_3(x,n_1)) \\ A_2 & = & \nu n_2.\mathsf{in}(y).\mathsf{out}(\mathsf{f}_2(y,w,n_2)) \end{array}$$

$\nu w.A_1$ and $\nu w.A_2$ resist against guessing attacks

A guessing attack on $\nu w.(A_1 \mid A_2)$:

$$\nu w.(A_1 \mid A_2) \to^* \nu w, \underbrace{n_1, n_2.(\{^{\mathsf{f}_1(w,n_1)}/_{x_1}\} \mid \{^{\mathsf{f}_2(\mathsf{f}_1(w,n_1),w,n_2)}/_{x_2}\} \mid \{^{w}/_{x_3}\})}_{\phi}$$

and $\nu w.\phi \mid \{^{w}/_{x}\} \not\approx \nu w.\phi \mid \nu w'.\{^{w'}/_{x}\}$

Well-taged protocols and composition

Intuitively, a protocol is well-tagged w.r.t. a secret w if all the occurrences of w are of the form $\mathsf{h}(\alpha, w)$

Definition (well-tagged)

M is α-tagged w.r.t. w if there exists M' s.t. $M'\{^{\mathsf{h}(\alpha,w)}/_{w}\} =_{\mathsf{E}} M$. A term is said well-tagged w.r.t. w if it is α-tagged for some name α.
A is α-tagged if any term occurring in it is α-tagged. An extended process is well-tagged if it is α-tagged for some name α.

Well-tagged processes compose!

Theorem (composition result)

Let A_1 be α-tagged and A_2 be β-tagged w.r.t. w.
If $\nu w.A_1$ and $\nu w.A_2$ are resistant to guessing attacks against w then $\nu w.(A_1 \mid A_2)$ is also resistant to guessing attacks against w.

Well-taged protocols and composition

Intuitively, a protocol is well-tagged w.r.t. a secret w if all the occurrences of w are of the form $\mathsf{h}(\alpha, w)$

Definition (well-tagged)

M is α-tagged w.r.t. w if there exists M' s.t. $M'\{^{\mathsf{h}(\alpha,w)}/_{w}\} =_{\mathsf{E}} M$. A term is said well-tagged w.r.t. w if it is α-tagged for some name α.
A is α-tagged if any term occurring in it is α-tagged. An extended process is well-tagged if it is α-tagged for some name α.

Well-tagged processes compose!

Theorem (composition result)

Let A_1 be α-tagged and A_2 be β-tagged w.r.t. w.
If $\nu w.A_1$ and $\nu w.A_2$ are resistant to guessing attacks against w then $\nu w.(A_1 \mid A_2)$ is also resistant to guessing attacks against w.

A secure transformation

Theorem

If $\nu w.A$ is resistant to guessing attacks against w then $\nu w.(A\{^{\mathsf{h}(\alpha,w)}/_{w}\})$ is also resistant to guessing attacks against w.

Easy, syntactic transformation: thumbrule for good design?

Remark on other transformations:

- replacing w by $\langle w, \alpha\rangle$ does not guarantee composition
- tagging encryptions (used in [CortierDelaitreDelaune07] to ensure composition of other properties) would add guessing attacks

Conclusion and future work

Passive guessing attacks do compose

Active guessing attacks do not compose in general

but for well-taged protocols

Secure transformation to obtain well-tagged protocols

Future work

Avoid tags : are there (interesting) classes of equational theories for which guessing attacks compose?

Other forms of composition :

- composition for observational equivalence
- sequential composition