

Simulation based security in the applied pi calculus

Stéphanie Delaune¹ Steve Kremer¹ Olivier Pereira²

¹LSV, ENS Cachan & CNRS & INRIA

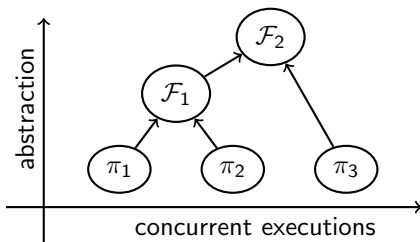
²Université Catholique de Louvain

CosyProofs 2010



Secure composition

Two dimensions:

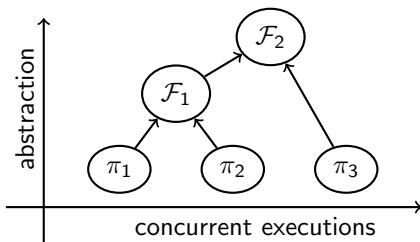


1. **Concurrency:** I proved π_1 secure.
Is it still secure when π_3 is also running?
2. **Abstraction:** I proved π_1 and π_2 to be good key exchange protocols.
Can I just abstract these as a device \mathcal{F}_1 providing keys?



Secure composition

Two dimensions:



1. **Concurrency:** I proved π_1 secure.
Is it still secure when π_3 is also running?
 2. **Abstraction:** I proved π_1 and π_2 to be good key exchange protocols.
Can I just abstract these as a device \mathcal{F}_1 providing keys?
- **Definitions:** Some protocols can have very natural definitions in terms of ideal functionalities (e.g., voting)



Can we just take standard crypto notions?

Question: Do computational composition and joint state theorems carry to the symbolic world?



Can we just take standard crypto notions?

Question: Do computational composition and joint state theorems carry to the symbolic world?

- ▶ No probabilities, no computational assumptions, life should be much easier!



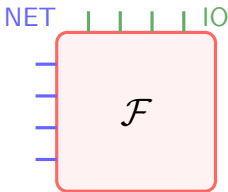
Can we just take standard crypto notions?

Question: Do computational composition and joint state theorems carry to the symbolic world?

- ▶ No probabilities, no computational assumptions, life should be much easier!
- ▶ Very different (and more natural) communication model: nondeterministic vs. sequential one



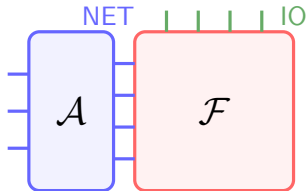
Composable security in the applied pi calculus



A functionality \mathcal{F} is a closed plain process with channel names in $\text{IO} \cup \text{NET}$



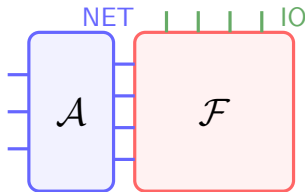
Composable security in the applied pi calculus



An adversary for \mathcal{F} is an evaluation context $\mathcal{A}[_]$ that closes the NET interface (not touching IO)



Composable security in the applied pi calculus

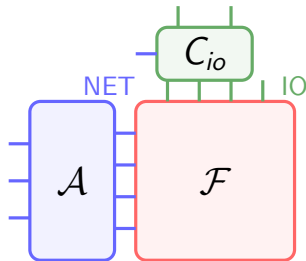


Remarks:

- ▶ If $\mathcal{A}_1[-]$ is an adversary for \mathcal{F} then $\mathcal{A}_1[\mathcal{F}]$ is a functionality
- ▶ If $\mathcal{A}_2[-]$ is an adversary for $\mathcal{A}_1[\mathcal{F}]$, then $\mathcal{A}_2[\mathcal{A}_1[-]]$ is an adversary for \mathcal{F}



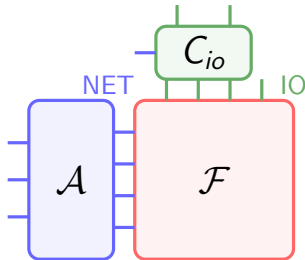
Composable security in the applied pi calculus



An IO context is an evaluation context $C_{io}[-]$ that can connect on the IO interface of \mathcal{F}



Composable security in the applied pi calculus



Remark:

- ▶ If $C_{io}[-]$ is an IO context for \mathcal{F} , then $C_{io}[\mathcal{F}]$ is a functionality.



Composable security...

A bunch of possible definitions:



Composable security...

A bunch of possible definitions:

- ▶ **Universally composable** (\leq^{UC}): $\forall \mathcal{A} \exists \mathcal{S} \mathcal{A}[\mathcal{F}_1] \stackrel{(?)}{\leq} \mathcal{S}[\mathcal{F}_2]$

where \mathcal{A} and \mathcal{S} are adversaries,



Composable security...

A bunch of possible definitions:

- ▶ Universally composable (\leq^{UC}): $\forall \mathcal{A} \exists \mathcal{S} \mathcal{A}[\mathcal{F}_1] \stackrel{(?)}{\leq} \mathcal{S}[\mathcal{F}_2]$
- ▶ UC with dummy adversary (\leq^{UCDA}): $\exists \mathcal{S} \mathcal{D}[\mathcal{F}_1] \stackrel{(?)}{\leq} \mathcal{S}[\mathcal{F}_2]$

where \mathcal{A} and \mathcal{S} are adversaries, and \mathcal{D} is the dummy adversary.



Composable security...

A bunch of possible definitions:

- ▶ Universally composable (\leq^{UC}): $\forall \mathcal{A} \exists \mathcal{S} \mathcal{A}[\mathcal{F}_1] \stackrel{(?)}{\preceq} \mathcal{S}[\mathcal{F}_2]$
- ▶ UC with dummy adversary (\leq^{UCDA}): $\exists \mathcal{S} \mathcal{D}[\mathcal{F}_1] \stackrel{(?)}{\preceq} \mathcal{S}[\mathcal{F}_2]$
- ▶ Black box (\leq^{BB}): $\exists \mathcal{S}. \forall \mathcal{A}. \mathcal{A}[\mathcal{F}_1] \stackrel{(?)}{\preceq} \mathcal{A}[\mathcal{S}[\mathcal{F}_2]]$

where \mathcal{A} and \mathcal{S} are adversaries, and \mathcal{D} is the dummy adversary.



Composable security...

A bunch of possible definitions:

- ▶ Universally composable (\leq^{UC}): $\forall \mathcal{A} \exists \mathcal{S} \mathcal{A}[\mathcal{F}_1] \stackrel{(?)}{\leq} \mathcal{S}[\mathcal{F}_2]$
- ▶ UC with dummy adversary (\leq^{UCDA}): $\exists \mathcal{S} \mathcal{D}[\mathcal{F}_1] \stackrel{(?)}{\leq} \mathcal{S}[\mathcal{F}_2]$
- ▶ Black box (\leq^{BB}): $\exists \mathcal{S}. \forall \mathcal{A}. \mathcal{A}[\mathcal{F}_1] \stackrel{(?)}{\leq} \mathcal{A}[\mathcal{S}[\mathcal{F}_2]]$
- ▶ Strong (\leq^{SS}): $\exists \mathcal{S}. \mathcal{F}_1 \stackrel{(?)}{\leq} \mathcal{S}[\mathcal{F}_2]$

where \mathcal{A} and \mathcal{S} are adversaries, and \mathcal{D} is the dummy adversary.



Composable security...

A bunch of possible definitions:

- ▶ Universally composable (\leq^{UC}): $\forall \mathcal{A} \exists \mathcal{S} \mathcal{A}[\mathcal{F}_1] \stackrel{(?)}{\leq} \mathcal{S}[\mathcal{F}_2]$
- ▶ UC with dummy adversary (\leq^{UCDA}): $\exists \mathcal{S} \mathcal{D}[\mathcal{F}_1] \stackrel{(?)}{\leq} \mathcal{S}[\mathcal{F}_2]$
- ▶ Black box (\leq^{BB}): $\exists \mathcal{S}. \forall \mathcal{A}. \mathcal{A}[\mathcal{F}_1] \stackrel{(?)}{\leq} \mathcal{A}[\mathcal{S}[\mathcal{F}_2]]$
- ▶ Strong (\leq^{SS}): $\exists \mathcal{S}. \mathcal{F}_1 \stackrel{(?)}{\leq} \mathcal{S}[\mathcal{F}_2]$

where \mathcal{A} and \mathcal{S} are adversaries, and \mathcal{D} is the dummy adversary.

A bunch of desirable properties:

- ▶ $\leq^{UCDA} \Rightarrow \leq^{UC}$
- ▶ or even $\leq^{UC} = \leq^{UCDA} = \leq^{BB} = \leq^{SS}$
- ▶ relations to be reflexive, transitive, composable, ...



What should we choose for $\stackrel{(?)}{\preceq}$?

How do we compare the two worlds?



What should we choose for $\stackrel{(?)}{\preceq}$?

How do we compare the two worlds?

1. Observational equivalence \approx /labelled bisimulation
Standard notion, lot of soundness results, tool support.



What should we choose for $\leq^{(?)}$?

How do we compare the two worlds?

1. Observational equivalence \approx /labelled bisimulation

Standard notion, lot of soundness results, tool support. But:

- ▶ \leq^{SS} does not seem reflexive (\mathcal{S} brings extra behaviors)
- ▶ \leq^{UCDA} does not seem to imply \leq^{UC}
(\mathcal{D} not transparent)
- ▶ transitivity also seems to fail



What should we choose for $\leq^?$

How do we compare the two worlds?

1. Observational equivalence \approx /labelled bisimulation
Standard notion, lot of soundness results, tool support. But:
 - ▶ \leq^{SS} does not seem reflexive (\mathcal{S} brings extra behaviors)
 - ▶ \leq^{UCDA} does not seem to imply \leq^{UC}
(\mathcal{D} not transparent)
 - ▶ transitivity also seems to fail
2. Observational preorder \leq /labeled simulation
Natural for refinement:
 - ▶ every behaviour on the left can be matched on the right



What should we choose for \leq ?^(?)

How do we compare the two worlds?

1. Observational equivalence \approx /labelled bisimulation

Standard notion, lot of soundness results, tool support. But:

- ▶ \leq^{SS} does not seem reflexive (\mathcal{S} brings extra behaviors)
- ▶ \leq^{UCDA} does not seem to imply \leq^{UC}
(\mathcal{D} not transparent)
- ▶ transitivity also seems to fail

2. Observational preorder \leq /labeled simulation

Natural for refinement:

- ▶ every behaviour on the left can be matched on the right

Makes everything work fine:

- ▶ reflexive, transitive, composable, security notions collapse



What should we choose for $\leq^?$

How do we compare the two worlds?

1. Observational equivalence \approx /labelled bisimulation

Standard notion, lot of soundness results, tool support. But:

- ▶ \leq^{SS} does not seem reflexive (\mathcal{S} brings extra behaviors)
- ▶ \leq^{UCDA} does not seem to imply \leq^{UC}
(\mathcal{D} not transparent)
- ▶ transitivity also seems to fail

2. Observational preorder \leq /labeled simulation

Natural for refinement:

- ▶ every behaviour on the left can be matched on the right

Makes everything work fine:

- ▶ reflexive, transitive, composable, security notions collapse

Questions:

- ▶ which properties are preserved, exactly, here?
- ▶ is there a subclass of processes for which \approx would be useful?



Composable security in the applied pi calculus

Strong simulatability: $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ iff \exists adv. $\mathcal{S} : \mathcal{F}_1 \preceq \mathcal{S}[\mathcal{F}_2]$



Composable security in the applied pi calculus

Strong simulatability: $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ iff \exists adv. $\mathcal{S} : \mathcal{F}_1 \preceq \mathcal{S}[\mathcal{F}_2]$

\leq^{SS} is a preorder:

- ▶ Reflexivity: $\mathcal{F} \leq^{\text{SS}} \mathcal{F}$
- ▶ Transitivity: If $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ and $\mathcal{F}_2 \leq^{\text{SS}} \mathcal{F}_3$ then $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_3$



Composable security in the applied pi calculus

Strong simulatability: $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ iff \exists adv. $\mathcal{S} : \mathcal{F}_1 \preceq \mathcal{S}[\mathcal{F}_2]$

\leq^{SS} is a preorder:

- ▶ Reflexivity: $\mathcal{F} \leq^{\text{SS}} \mathcal{F}$
- ▶ Transitivity: If $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ and $\mathcal{F}_2 \leq^{\text{SS}} \mathcal{F}_3$ then $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_3$

\leq^{SS} is closed under application of IO-context:

- ▶ If $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ then $C_{io}[\mathcal{F}_1] \leq^{\text{SS}} C_{io}[\mathcal{F}_2]$



Composable security in the applied pi calculus

Strong simulatability: $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ iff \exists adv. $\mathcal{S} : \mathcal{F}_1 \preceq \mathcal{S}[\mathcal{F}_2]$

\leq^{SS} is a preorder:

- ▶ Reflexivity: $\mathcal{F} \leq^{\text{SS}} \mathcal{F}$
- ▶ Transitivity: If $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ and $\mathcal{F}_2 \leq^{\text{SS}} \mathcal{F}_3$ then $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_3$

\leq^{SS} is closed under application of IO-context:

- ▶ If $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ then $C_{io}[\mathcal{F}_1] \leq^{\text{SS}} C_{io}[\mathcal{F}_2]$

\leq^{SS} is preserved under composition

- ▶ If $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ then $\mathcal{F}_1 \mid \mathcal{F}_3 \leq^{\text{SS}} \mathcal{F}_2 \mid \mathcal{F}_3$
- ▶ If $\mathcal{F}_1 \leq^{\text{SS}} \mathcal{F}_2$ then $!\mathcal{F}_1 \leq^{\text{SS}} !\mathcal{F}_2$



Application: mutual authentication

or: what about Needham-Schroeder-Lowe?



Application: mutual authentication

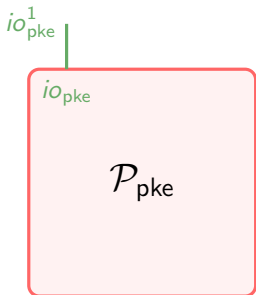
or: what about Needham-Schroeder-Lowe?

Steps:

1. Functionality for public key encryption
2. Joint state theorem for public key encryption (multiple sessions)
3. Functionality for mutual authentication
4. Design simulator for NSL and prove preorder relation



A public key encryption service

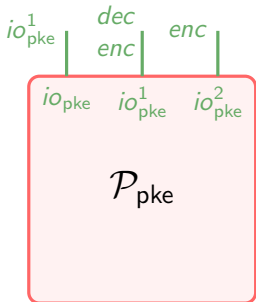


\mathcal{P}_{pke} functionality: usual symbolic encryption service

- ▶ Init: private channel io_{pke}^1 transmitted through io_{pke}
 \mathcal{P}_{pke} generates private key sk and sends $pub(sk)$ back
- ▶ Encryption goes through io_{pke}^1 and io_{pke}^2
- ▶ Decryption goes through io_{pke}^1 only



A public key encryption service

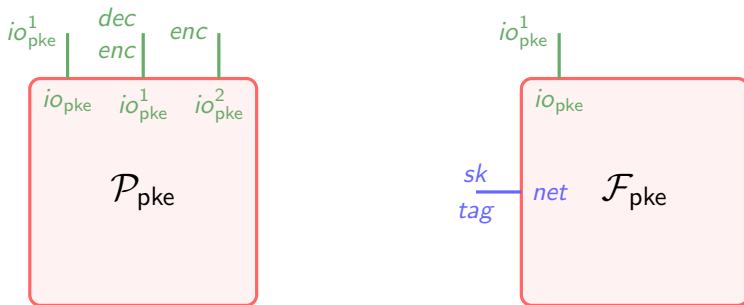


\mathcal{P}_{pke} functionality: usual symbolic encryption service

- ▶ Init: private channel io_{pke}^1 transmitted through io_{pke}
 \mathcal{P}_{pke} generates private key sk and sends $pub(sk)$ back
- ▶ Encryption goes through io_{pke}^1 and io_{pke}^2
- ▶ Decryption goes through io_{pke}^1 only



A public key encryption service

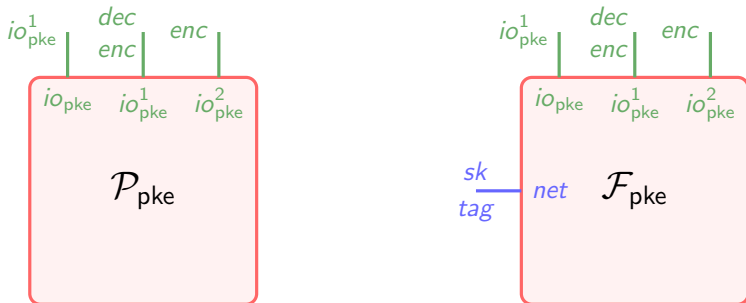


\mathcal{F}_{pke} functionality: ideal encryption service

- ▶ Confidentiality in \mathcal{P}_{pke} depends of $pub(sk)$
- ▶ Multiple session case treated by saying “we do not care about the key”
- ▶ Ideal encryption service secure independently of the key



A public key encryption service

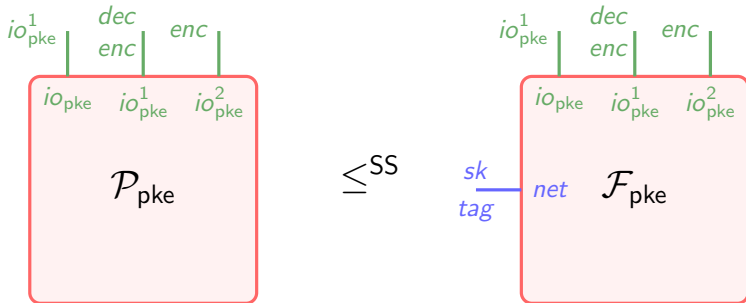


\mathcal{F}_{pke} functionality: same as \mathcal{P}_{pke} except:

- ▶ \mathcal{F}_{pke} takes sk and a tag from the adversary
- ▶ \mathcal{F}_{pke} uses two layers of encryption, with internal ssk key (corresponding public key never goes out)
 $enc'(m, pub(sk)) := enc(tag, enc(m, pub(ssk)), pub(sk))$



A public key encryption service

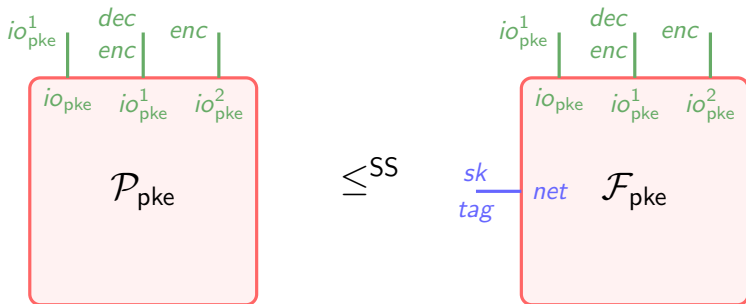


We have:

$$\triangleright \mathcal{P}_{\text{pke}} \leq^{SS} \mathcal{F}_{\text{pke}}$$



A public key encryption service



We have:

$$\triangleright \mathcal{P}_{\text{pke}} \leq^{SS} \mathcal{F}_{\text{pke}}$$

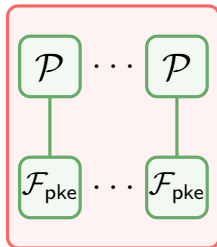
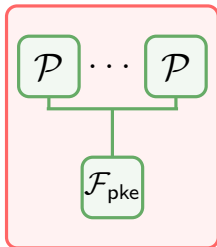
From composition:

$$\triangleright \mathcal{P}_{\text{pke}} \leq^{SS} \mathcal{F}_{\text{pke}} \Rightarrow !\mathcal{P}_{\text{pke}} \leq^{SS} !\mathcal{F}_{\text{pke}}$$

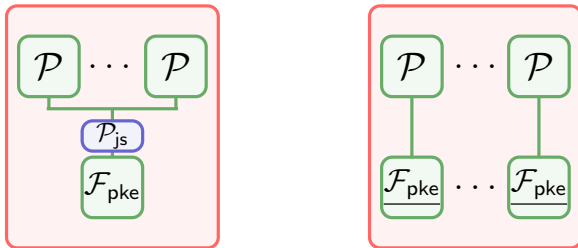
But this means that each session uses its own functionality/keys!



Composition with joint state



Composition with joint state

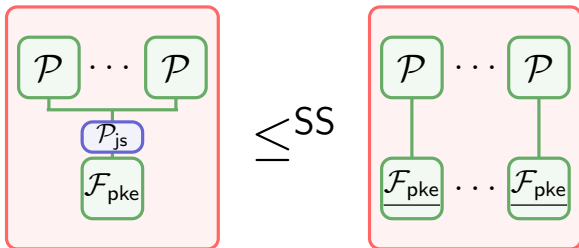


Main elements:

- ▶ \mathcal{P}_{js} adds and removes sids to requests
- ▶ \mathcal{F}_{pke} is \mathcal{F}_{pke} with sids added in messages



Composition with joint state

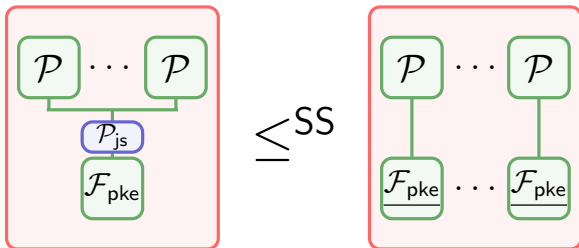


Joint state theorem:

▶ $\mathcal{P}_{js}[\mathcal{F}_{pke}] \leq^{SS} \underline{\mathcal{F}_{pke}}$



Composition with joint state



Joint state theorem:

$$\triangleright \mathcal{P}_{js}[\mathcal{F}_{pke}] \leq^{SS} \underline{\mathcal{F}_{pke}}$$

Using properties of \leq^{SS}

$$\triangleright \mathcal{P}_{js}[\mathcal{P}_{pke}] \leq^{SS} \mathcal{P}_{js}[\mathcal{F}_{pke}] \leq^{SS} \underline{\mathcal{F}_{pke}} \quad (\text{But not } \mathcal{P}_{js}[\mathcal{P}_{pke}] \leq^{SS} \underline{\mathcal{P}_{pke}})$$



Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality

$\mathcal{F}_{\text{init}}$

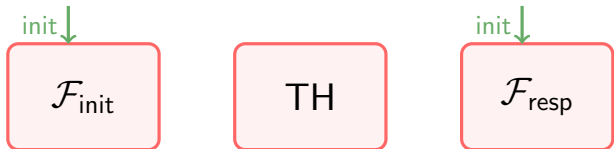
TH

$\mathcal{F}_{\text{resp}}$



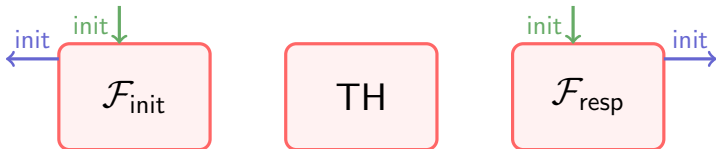
Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality



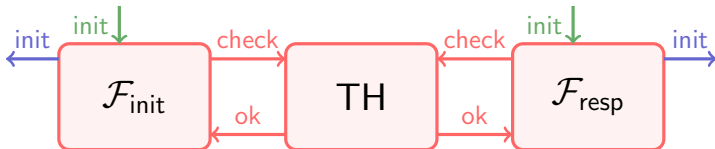
Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality



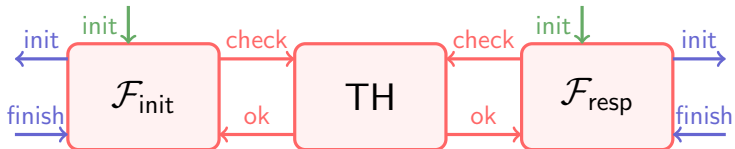
Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality



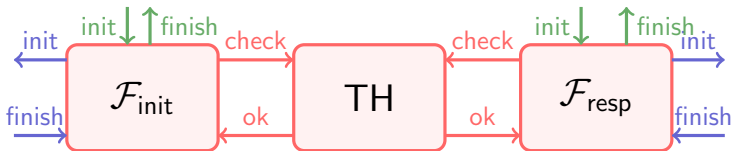
Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality



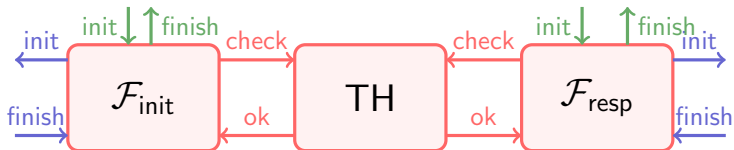
Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality



Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality

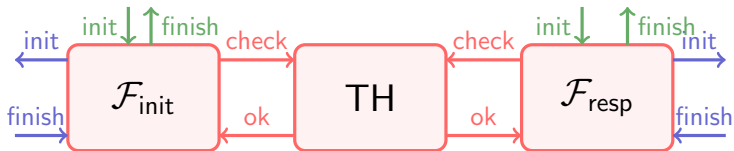


Distributed implementation based on the NSL protocol



Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality

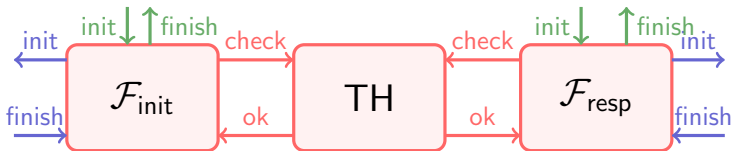


Distributed implementation based on the NSL protocol



Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality

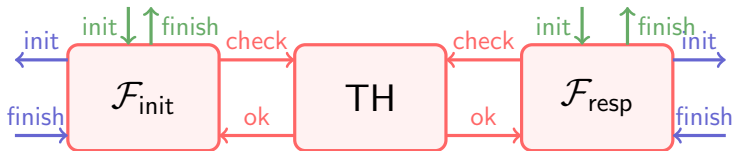


Distributed implementation based on the NSL protocol



Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality

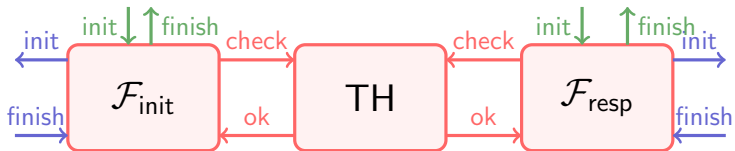


Distributed implementation based on the NSL protocol

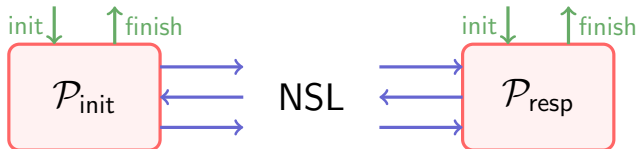


Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality

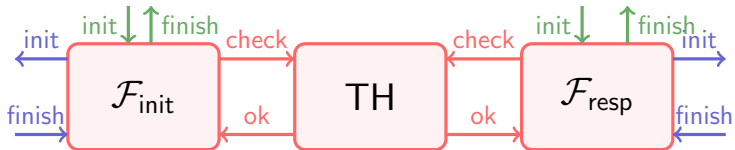


Distributed implementation based on the NSL protocol

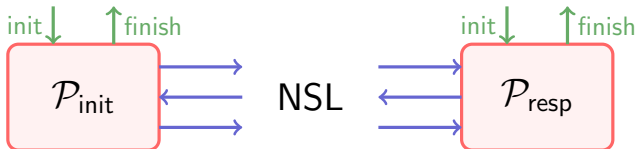


Application: Mutual authentication functionality

Definition of an ideal mutual authentication functionality



Distributed implementation based on the NSL protocol



Proof:

- ▶ Simulator executes NSL protocol when receiving 'init' and sends back 'finish' when protocol succeeds
- ▶ Use joint state theorem to go "from one session to many"



Conclusion

We obtain:

- ▶ Framework for hierarchical reasoning in a Dolev-Yao like model
- ▶ Applications: asymmetric encryption, joint state, mutual authentication



Conclusion

We obtain:

- ▶ Framework for hierarchical reasoning in a Dolev-Yao like model
- ▶ Applications: asymmetric encryption, joint state, mutual authentication

Further works:

- ▶ Better understand the properties of \leq^{SS} (and \preceq vs. \approx)
- ▶ Show computational soundness of symbolic proofs of \leq^{SS}
- ▶ Ideal functionalities look appealing for multiparty protocols, e.g., electronic voting



Conclusion

We obtain:

- ▶ Framework for hierarchical reasoning in a Dolev-Yao like model
- ▶ Applications: asymmetric encryption, joint state, mutual authentication

Further works:

- ▶ Better understand the properties of \leq^{SS} (and \preceq vs. \approx)
- ▶ Show computational soundness of symbolic proofs of \leq^{SS}
- ▶ Ideal functionalities look appealing for multiparty protocols, e.g., electronic voting

S. Delaune, S. Kremer and O. Pereira. Simulation based security in the applied pi calculus. Research Report 2009/267, Cryptology ePrint Archive, 2009.



Thank you!

