Stateful Protocol Composition
or:
How we accidentally did
Sequential Composition

Andreas V. Hess (DTU)
Sebastian A. Mödersheim (DTU)
Achim D. Brucker (University of Sheffield)

This work was supported by the
DFF Sapere Aude project CompoSec.

ÖSD 2018
Example: Problem with Parallel Composition

**NSL**

- $\{NA, A\}_{pk(B)}$
- $\{NA, NB, B\}_{pk(A)}$
- $\{NB\}_{pk(B)}$

**Quatsch**

- $\{NA, A\}_{pk(B)}$
- $\{NA, NB, B\}_{pk(A)}$
- $\{NB\}_{pk(B)}$

Quatsch
Example: Inserting Tags

**NSL**

- \{\textit{nsl}, \textit{NA}, \textit{A}\}_\text{pk(B)} \rightarrow \{\textit{nsl}, \textit{NA}, \textit{NB}, \textit{B}\}_\text{pk(A)}
- \{\textit{nsl}, \textit{NB}\}_\text{pk(B)}
- \{\textit{nsl}, \textit{NA}, \textit{A}\}_\text{pk(B)} \rightarrow \textit{NA}

**Quatsch**

- \{\textit{quatsch}, \textit{NA}, \textit{A}\}_\text{pk(B)}
Example: Formats

- Transparent functions $f_1, \ldots, f_5$ as abstract syntax
- Concrete syntax must be unambiguous and pairwise disjoint
- Result: Then this it is sound to consider abstract syntax.
Assign intended types like $A : Agent$, $NA : Nonce$

SMP: Patterns of the protocol messages (plus subterms, instances)

Requirement: $s, t \in SMP \setminus \mathcal{V}$ may have a unifier only if they have the same type.

Result: if there is an attack, then there is well-typed attack
Parallel Composition

**Requirement:** the message patterns (SMP) of the protocols are disjoint

- No protocol leaks long-term secrets to the intruder

- Result: if there is an attack on the composition, then there is one on the individual protocols
The Proof Argument

- Attack trace as a sequence of messages that the intruder sends/receives.
- The subtraces for the two protocols work on their own.
- Problem with this argument: often the intruder may use in a $P_1$ step some messages he learned from a $P_2$ step.
The Proof Argument (Example)

- Attack trace as a sequence of messages that the intruder sends/receives
- The subtraces for the two protocols work on their own.
- Problem with this argument: often the intruder may use in a $P_1$ step some messages he learned from a $P_2$ step.
• Constraints – **symbolic** attack traces
• Describing a set of solutions
• Question: is there one solution where all messages are **homogeneous** (not mixing the two protocols)?
• Constraints – *symbolic* attack traces
• Describing a set of solutions
• Question: is there one solution where all messages are *homogeneous* (not mixing the two protocols)?
- Constraint reduction: a complete way to find all solutions of a constraint
- If the protocols have disjoint SMPs, the procedure will never unify messages from the two protocols
- The remaining variables represent choices of the intruder and there is always a homogeneous one.
## zeb’s Keyserver

### Registration($U : User, S : Server, PK : Value$)

\[
\begin{align*}
  U/S : & \text{ create}(PK) \\
         & \text{ insert}(PK, ring(U)) \\
         & \text{ insert}(PK, db(S, U, valid)) \\
  U/S \rightarrow_\_ PK
\end{align*}
\]

### Update($U : User, S : Server, PK : Value, NPK : Value$)

\[
\begin{align*}
  U : & \text{ select}(PK, ring(U)) \\
       & \text{ delete}(PK, ring(U)) \\
       & \text{ create}(NPK) \\
       & \text{ insert}(NPK, ring(U)) \\
  U \rightarrow S : & \text{ sign}(inv(PK), NPK) \\
  S : & \text{ in}(PK, db(S, U, valid)) \\
       & \text{ notin}(NPK, db(S, \_ \_ \_)) \\
       & \text{ delete}(PK, db(S, U, valid)) \\
       & \text{ insert}(PK, db(S, U, revoked)) \\
       & \text{ insert}(NPK, db(S, U, valid)) \\
  S \rightarrow_\_ : & \text{ inv}(PK)
\end{align*}
\]
Extension to Stateful Protocols

- Idea: set operations (insert, delete, positive/negative checks) can be encoded in constraints with inequalities
- Typing result still works for a large class of problems
- Compositionality also relatively easily as long as
  - The composed protocols do not share the database
  - All secrets are persistent (e.g. not revealing private key after revocation)
  - Can we do without these restrictions?
Another Protocol that uses the database

PasswordBasedRegistration($U : User, S : Server, PK : Value$)

$S \rightarrow U : \text{N}$
- $U : \text{create}(PK)$

$U \rightarrow S : \{\text{pw}(U, S), N, PK\}_{pk(B)}$
- $S : \text{insert}(PK, \text{db}(S, U, \text{valid}))$

$S \rightarrow _{} : \text{PK}$

- This protocol inserts into the same database.
- Both protocols do ensure authentication and freshness of the key and do not leak it while a key is in the valid set.
- Should be ok to compose like this!
Idea: Protocol Abstraction

- Problem: due to shared use of databases we cannot split the constraints into individual protocols anymore.
- Idea: abstract each protocol into the modifications it can make to the shared database.

**zeb’s keyserver abstraction**

★ For honest agents $A$: generate fresh key and insert into valid database of $A$.

★ For intruder: insert any intruder-known key into valid database for $i$.

★ Revocation: delete a key from valid and give private key to the intruder.

- Verify each protocol individually but together with the abstraction of the other protocol.

★ $P_1 \parallel P_2^*$

★ $P_1^* \parallel P_2$

★ i.e., taking into account the changes the other protocol can make.
Stateful Protocols

Stateful Parallel Composition

- Protocols can share the database
- Abstract the modifications each protocol can make and take into account in the other.
  - ★ Assume–guarantee reasoning: e.g. inserted keys are authenticated.
- Declassification of secrets possible: just coordinate via a shared set (e.g. valid)
How we accidentally did Sequential Composition

Scenario

- \( P_1 \) generates and exchanges a shared key (e.g. TLS handshake)
- \( P_2 \) uses the shared key to start with (e.g. Application protocol with TLS transport)
- Under which conditions is the composition secure?

- Basically, a special case of Stateful Parallel Composition where
  - \( P_1 \) inserts the generated key into a shared set \( keys(A, B) \)
  - \( P_2 \) consumes keys from there.
Instead of a Conclusion

- Core result formalized in Isabelle:
  - ★ Typing result (CSF 2017)
  - ★ Typing result for stateful protocols (CSF 2018)
  - ★ Stateful composition result on the constraint level (submitted)
- Revealed several mistakes in existing composition proofs
- The clean formalization has lead to simplifications and generalizations
- Applicable as a theorem in Isabelle:
  - ★ Prove the security of the protocols individually in the typed model
  - ★ Apply our result to get the proof for the untyped model and their composition