A Calculus of Challenges and Responses

Matteo Maffei

---

Saarland University

Joint work with:

Michael Backes (Max Planck Institute for Software Systems)
Agostino Cortesi (Venice University)
Riccardo Focardi (Venice University)
Formal methods for security protocols

* Model-checking and theorem-proving:
  - Precise, attacks easily derivable
  - No *termination* guarantee, not compositional

* Type systems:
  - Guaranteed *termination, compositional*
  - Useful insights on *why* a protocol is correct
  - Restrictive syntactic patterns, refining the analysis means proving the soundness from scratch
  - False positives hard to detect, attacks hard to derive
**Challenge-response implementations**

\[
\begin{align*}
A & \quad B \\
\leftarrow & \quad n \\& \quad \{B,n,m\}_{k_A}^\rightarrow \\
A & \quad B \\
\leftarrow & \quad n \\
& \quad \{n,\{m\}_{k_B}^+\}_{k_A}^\rightarrow \\
A & \quad B \\
\leftarrow & \quad n \\
& \quad \{B,n,\{m\}_{k_B}^+\}_{k_A}^\rightarrow \\
A & \quad B \\
\leftarrow & \quad n \\
& \quad m,\{B,n,h(m)\}_{k_A}^\rightarrow
\end{align*}
\]

*Intuitively*: public challenge, signed response

*Idea*: abstract away from the specific cryptographic implementation and focus on the challenge-response scheme.
Analysis technique

- Specify the protocol in $\rho$-spi calculus

- **Abstract** the protocol specification into **CR calculus**
  - cryptography is abstracted into challenges and responses, each of them associated to a security level

- Check the protocol specification by an *effect system*

- The **safety** of the abstract protocol implies the safety of its $\rho$-spi implementations
\textbf{\( \rho \)-spi calculus}

\begin{align*}
\text{Names} & \\
\text{ } & \text{Terms} \\
a ::= & \ n, m \quad \text{(Msg)} & T ::= & \ a \quad \text{(Name)} \\
I, J, A, B, E & \quad \text{(Id)} & [?]x, y, z & \quad \text{(Var)} \\
\text{Keys} & & \text{Tag}(T) & \quad \text{(Tag)} \\
k ::= & \ k_{IJ} \quad \text{(Sym)} & (T_1, T_2) & \quad \text{(Pair)} \\
k^+_I, k^-_I & \quad \text{(Asym)} & \{ T \}_K & \quad \text{(Enc)} \\
\text{Proc} & & h(T) & \quad \text{(Hash)} \\
P, Q ::= & \new(n).P & \text{MAC}_K(T) & \quad \text{(MAC)} \\
\text{ } & \in(T).P & & \\
\text{ } & \out(T).P & & \\
\text{ } & \begin{begin}_N(A, I, M_1; M_2).P & & \\
\text{ } & \begin{end}_N(A, I, M_1; M_2).P & & \\
A \triangleright P & & \text{(Begin)} & \\
P|Q & & \text{(End)} & \\
!P & & \text{(Princ)} & \\
\emptyset & & \text{(Par)} & \\
\text{ } & & \text{(Repl)} & \\
\text{ } & & \text{(Stop)} & \\
\end{align*}

\begin{itemize}
\item \textit{Input with pattern-matching}
\item \textit{K} ranges over keys and vars
\item In begin-end assertions, \( M_1 \) and \( M_2 \) are tuples of names and variables sent in the challenge and response, respectively.
\end{itemize}
A simple protocol in $\rho$-spi calculus

\[
\begin{array}{c}
A \\
\quad n \\
\quad m,\{B,n,h(m)\} \xrightarrow[k_A]{-} \\
B
\end{array}
\]

\[
!A \triangleright \text{in(?x).} \\
\quad \text{new}(m). \\
\quad \text{begin}_x(A,B;m). \\
\quad \text{out}(m,\{B,x,h(m)\})
\]

\[
!B \triangleright \text{new}(n). \\
\quad \text{out}(n). \\
\quad \text{in(?y, \{B,n,h(y)\})}\xrightarrow[k_A]{-} \\
\quad \text{end}_n(B,A;y)
\]
Abstracting away from cryptography

- Ciphertexts are abstracted into challenges and responses, each of them associated to a security level
- A security lattice describes a partial order relation on security levels
CR calculus

(CR Names) $a ::= n, m$ (Msg)
$I, J, A, B, E$ (Id)
$\top$ (Any)
$\bot$ (Failure)

(CR Terms) $T ::= a$ (Name)
$[?]x, y, z$ (Var)
$(T_1, T_2)$ (Pair)
$C_{N}^{(\ell, \ell')} (I, J, M)$ (Chal)
$R_{N}^{(\ell', \ell)} (I, J, M)$ (Resp)

Notation: $\ell \in \{\text{Pub, Tnt, Priv, Int}\}$
$M, N$ denote tuples of names and vars
A simple protocol in CR calculus

\[
\begin{align*}
\text{!} & \text{A} \triangleright \text{in}(\text{Chal}_n^{\text{Pub}, \text{Int}}(\text{B, A})). \quad \text{new}(m). \\
& \quad \text{begin}_x(\text{A, B; m}). \quad \text{out}(m, \text{Resp}_x^{\text{Pub}, \text{Int}}(\text{A, B, m})). \quad | \quad \text{!} \text{B} \triangleright \text{new}(n). \\
& \quad \quad \quad \text{out}(\text{Chal}_n^{\text{Pub}, \text{Int}}(\text{B, A})). \quad \text{in}(?y, \text{Resp}_n^{\text{Pub}, \text{Int}}(\text{A, B, y})). \\
& \quad \quad \quad \text{end}_n(\text{B, A; y})
\end{align*}
\]
Trace-based semantics (intuitively)

\[
\begin{array}{c}
\text{ENCRIPTION} \\
\frac{s \vdash G \quad s \vdash k}{s \vdash \{G\}_k}
\end{array}
\quad \begin{array}{c}
\text{DECRIPTION} \\
\frac{s \vdash \{G\}_k \quad s \vdash \overline{k}}{s \vdash G}
\end{array}
\]

* Dolev-Yao encryption and decryption in p-spi calculus

* Processes can input any term known to the environment and matching the input pattern
Trace-based semantics (intuitively)

WRITE

\[
\begin{align*}
  s \vdash G_1, G_2 & \quad \ell \leq \text{Tnt} \\
  s \vdash C_{G_1}^{\ell, \ell'} (I, J, G_2) & \quad s \vdash R_{G_1}^{\ell', \ell} (I, J, G_2)
\end{align*}
\]

READ

\[
\begin{align*}
  s \vdash C_{G_1}^{\ell, \ell'} (I, J, G_2) & \lor s \vdash R_{G_1}^{\ell', \ell} (I, J, G_2) \\
  \ell \leq \text{Int} & \\
  s \vdash G_1, G_2
\end{align*}
\]

* The corresponding rules in CR calculus

A Calculus of Challenges and Responses
Cryptography abstraction

- Abstraction parameterized by a function, called *encryption abstraction*, which abstracts ciphertexts and MACs into challenge and response terms.
- Encryption abstractions defined and possibly extended by the user
- Extensions to additional protocol classes enjoy a *soundness theorem* provided that these extensions satisfy certain explicit, easily checkable *conditions*
Abstraction of terms

\[ \alpha(a) = a \quad \alpha((T_1, T_2)) = (\alpha(T_1), \alpha(T_2)) \]
\[ \alpha(x) = x \quad \alpha(\text{Tag}(T')) = \alpha(T') \]
\[ \alpha(?x) = ?x \quad \alpha(h\{ T \}) = \alpha(T) \]

\[ \alpha(\{ T \}_K) = \begin{cases} 
\bar{f}(\{ T \}_K) & \text{if } \{ T \}_K \in \text{dom}(f) \\
\bar{\alpha}(T) & \text{if } \{ T \}_K \notin \text{dom}(f) \land \\
\n & \# \text{ valid } \sigma \text{ and } R \in \text{dom}(f) \\
\bot & \text{s.t. } \llbracket \{ T \}_K \rrbracket \sigma = \llbracket R \rrbracket \\
\end{cases} \]

\[ \ast \text{ } f \text{ is a the closure of } f, \text{ which is a } \text{partial function} \text{ from} \]
ciphertexts and MACs to abstract terms

\[ \ast \text{ Similar abstraction for MACs} \]
Sufficient conditions (intuitively)

\[ f : \{ \{ T \} \}_k \mid MAC_k(T) \rightarrow \text{Chal}_N^\ell(I, J, M) \mid Resp_N^\ell(I, J, M) \]
is an encryption abstraction \( \text{iff} \)

\begin{itemize}
  \item \textit{Scope} \quad \text{vars}(T) = \text{vars}(N) \cup \text{vars}(M)
  
  \item \textit{Un.Abs.} \quad \text{if} \ \exists T, T' \in \text{dom}(f), \sigma, \sigma' \ \text{s.t.} \ [[T\sigma]] = T'\sigma' \ \text{then} \ T = T'
  
  \item \textit{Keys} \quad \text{either} \ k = k^+ \ \text{and} \ \ell \leq \text{Tainted}
      \text{or} \ k = k^- \ \text{and} \ \ell \leq \text{Int}
      \text{or} \ k = k_{AB}
\end{itemize}
Sound abstraction

\[ f(\{B, x, z\}_{k_{AB}}) = \text{Resp}^\text{Pub,Priv}_X(A, B, z) \]

is an encryption abstraction for the following protocol:

\[
\begin{array}{cccc}
A & B & A & B \\
\leftarrow & n & \rightarrow & \leftarrow \\
\{B, n, m\}_{k_{AB}} & \text{Chal}^\text{Pub,Priv}_n(B, A) & \text{Resp}^\text{Pub,Priv}_n(A, B, m) & \rightarrow \\
\end{array}
\]
Unsound abstraction

Each protocol can be abstracted as before...

\[ f_1(\{ B, x, z \}_{k_{AB}}) = \text{Resp}^{\text{Pub},\text{Priv}}_X (A, B, z) \]

\[ f_2(\{ B, x, z \}_{k_{AB}}) = \text{Resp}^{\text{Pub},\text{Priv}}_X (B, A, z) \]

...but the union of these functions is not a function and thus it is not an encryption abstraction!
Unsound abstraction

* The two protocols are safe when independently executed but they are flawed when executed within the same environment:

\[ A \xrightarrow{n} E(B) \]

\[ \leftarrow \quad \{B,m,n\}_{k_{AB}} \xrightarrow{n} \quad \{B,m,n\}_{k_{AB}} \]
A Calculus of Challenges and Responses


Sound abstraction

We can define a valid encryption abstraction after tagging the two protocols:

\[ f(\{\text{Verif}(B), x, z\}_{k_{AB}}) = \text{Resp}^{\text{Pub},\text{Priv}}_{X}(A, B, z) \]
\[ f(\{\text{Claim}(B), x, z\}_{k_{AB}}) = \text{Resp}^{\text{Pub},\text{Priv}}_{X}(B, A, z) \]

\[ \star \text{ is an encryption abstraction} \]
Soundness

★ *Theorem (Soundness)* If \(<s,P> \rightarrow <s',P'>\) then

\(<\alpha(s),\alpha(P)> \rightarrow <\alpha(s'),\alpha(P')>\)

★ Both p-spi calculus and CR calculus have a trace-based semantics

★ The abstraction of *traces and processes* is given by the abstraction of the terms occurring therein

★ This abstraction is preserved by process reduction
Safety

Definition (Safety) A trace $s$ is safe if every $\text{end}_N(A,B,M_1;M_2)$ occurring therein is preceded by a distinct $\text{begin}_N(B,A,M_1;M_2)$. A process $P$ is safe if all the traces generated by $P$ are safe.

The definition of safety for traces and processes in CR calculus is similar.
Static analysis

✿ The abstraction does not deal with the multiplicity of protocol sessions: authenticity is still undecidable

✿ Abstract protocols are statically analyzed by an effect system

✿ Abstraction and effect system are fully independent

✧ A refinement of the abstraction does not affect the soundness of the static analysis and vice-versa
Effect system (intuitively)

★ Effects track the actions performed by processes

★ An end event must be preceded by the output of a corresponding challenge and the input of a corresponding response

★ A begin event must be preceded by the input of a corresponding challenge and followed by the output of the corresponding response

★ The analysis is compositional and automated
Effect system (intuitively)

BEGIN

\[ \vdash P : e + [!R_{N}^{C,R}(A, I, M_2)] \quad M_2 \text{ ground} \]

\[ \vdash \text{begin}_{N}(A, I, M_1, M_2).P : e + [?C_{N}^{C,R}(I, A, M_1)] \]

END

\[ \vdash P : e \quad M_1, \text{ ground} \]

\[ \vdash \text{end}_{n}(A, I, M_1, M_2).P : e + 
[!C_{n}^{C,R}(A, I, M_1), ?R_{n}^{C,R}(I, A, M_2), \text{fresh}(n)] \]
Safety results

- **Theorem (Abstract Safety)** If $\vdash P : []$, then $P$ is safe
- **Theorem (Safety)** If $\vdash \alpha(P) : []$, then $P$ is safe
- **Theorem (Compositionality)** $\vdash P_1 \mid \ldots \mid P_n : []$ if and only if $\forall i \in [1,n], \vdash P_i : []$
Other results

- Our framework deals also with protocols for:
  - **Session-key distribution**: session keys are first authenticated and then abstracted similarly to long-term symmetric keys
  - **Mutual authentication**: ciphertexts may represent both challenges and responses
Conclusions

★ Contribution: abstraction of authentication protocols

- A single proof of safety suffices to prove the safety of protocols sharing the same abstraction
- Soundness of extensions to additional protocols guaranteed by easily checkable conditions
- Compositional static analysis of authenticity
- Analyzed several protocols out of SPORE

★ Future work:

- Analysis of cryptographic protocol implementations
- Computational soundness results