Pragmatic Guidelines for Formal Modeling of Security Ceremonies

Barbara Fila, Ermenda Hoxha

Pragmatic guidelines for formal modeling of security ceremonies

Barbara Fila[1,2][0000−0002−1824−7621] and Ermenda Hoxha[1,3][0009−0002−5144−1933]

[1] IRISA, Rennes, France
[2] INSA Rennes, France
[3] Politecnico di Milano, Italy
barbara.fila@irisa.fr, ermenda.hoxha@mail.polimi.it

Abstract. Formal modeling and verification is a powerful and widely adopted tool in the field of cybersecurity. It consists of devising and analyzing abstract models of real-life systems or situations to be evaluated. Creating accurate formal models that faithfully represent reality is crucial for performing meaningful analysis. Due to the diversity of aspects – technical, physical, human, and environmental – that may impact ceremony execution, formal modeling of ceremonies is a challenging task. The objective of our work is to scrutinize an existing formal framework for ceremony modeling and assess its ability to capture unexpected situations caused by faults, errors, negligence, or misbehavior. We analyze the framework's building components and formulate pragmatic guidelines to assist modelers in designing meaningful, rich ceremony models. We validate our findings on a two-factor authentication case study.

Keywords: Ceremony · Formal modeling · Faulty behavior · Erroneous behavior · Malicious behavior · Unexpected scenario · Attack trace

1 Introduction

The exponential progress of technology over the past two decades has led to a society powered by digital interactions. However, behind the glossy surface of such modern world lie significant security challenges that need to be properly addressed to protect not only the technologies and digital assets, but above all, the people who rely on them. Governmental and industry reports show that humans are the weakest link in the cybersecurity chain. Verizon's yearly report [15] attributes roughly 60% of data breaches to human error, misunderstanding, or manipulation. World Economic Forum reports in [16] that phishing and social engineering attacks rose dramatically in 2024, affecting around 42% of surveyed organizations. Worryingly, technological solutions are ubiquitous even among users with little understanding of how they work. The FBI's Internet Crime Complaint Center has received more than 9 million complaints over the past five years globally, with people over 50 being disproportionately affected [12].

To include human and other previously considered out-of-band components into the security analysis of distributed interactions, Ellison proposed in [8] to extend communication protocols to *ceremonies* where human beings, user–interface

interactions, human-to-human communications, and the transfer of physical artifacts carrying data are subject to the same reasoning as any other element of a communication protocol. Ellison argues that inclusion of humans and out-of-band components in the formal model leads to systems that are potentially secure against both technical attacks and social engineering. Formal modeling of humans and their environment is however a challenging task. Humans have limited computational skills and non-deterministic behavior, and their decisions may be affected by the environment, emotions, and chance.

To address the complex task of placing humans and all the out-of-band elements they bring alongside other technological entities, Fila and Radomirović introduced in [10] a rigorous formalism for ceremony specification and analysis, that we refer to in our article as *the ceremony modeling framework* (*the CM framework*, for short). The generic aspect of this formalism makes it possible to formalize humans, physical objects, locations, emotions using the same constructs as for modeling computing processes. The approach is however highly theoretical, and its practical adoption could benefit from clear guidelines explaining how to properly use its features to design useful ceremony models.

In the present work, *we review the CM framework with the goal of devising pragmatic heuristics for its practical usage*. We are particularly interested in the framework's expressive power regarding modeling of unexpected scenarios resulting from faulty, erroneous or malicious behavior of ceremony participants.

- We begin by analyzing the formal – syntactical and semantical – building blocks of the CM framework and *identify the constraints* they impose, limiting the possibility of capturing unexpected situations.
- With the help of a simple toy ceremony, we *illustrate the impact of each such constraint* on the expressiveness of a ceremony model.
- We then propose *targeted modeling strategies* that spell out how to overcome the identified limitations and grasp the desired behavior.
- We finally *validate* the suggested modeling strategies *on a larger case study* by modeling and analyzing a realistic two-factor authentication ceremony.

In Section 2, we review relevant existing work. Section 3 is devoted to the presentation of the CM framework introduced in [10]. The CM framework analysis, its limitations and the modeling guidelines that we have developed are presented in Section 4. Section 5 briefly describes our two-factor authentication case study. Section 6 concludes the paper and draws directions for future investigations.

2 Related work

The concept of *ceremonies*, introduced by Carl Ellison in 2007 [8], marks a critical shift in distributed communication analysis. Ellison argues that ceremonies are a superset of protocols that account for all previously out-of-band elements idealized or ignored by protocol verification methods.

Inspired by [8], Bella and Coles-Kemp advocate the value of *formal modeling of ceremonies* aiming at their rigorous understanding and analysis. In [3], they

break down a ceremony into layers, ranging from network processes to societal interactions, and design a formal approach to study interactions between user persona and computer interface.

In parallel, Carlos et al. propose a fully formal model for human knowledge distribution in security ceremonies, and investigate communication over human–human and human–interface channels [5]. Their work paves the way for tailored *threat models* and verification that considers also *contextual factors*. In a follow-up work [6], Carlos et al. argue that the classic Dolev–Yao adversary [7] is overly powerful and rarely realistic for real-world ceremonies. They propose to start from the standard Dolev–Yao model and selectively remove attacker capabilities depending on a specific context, yielding a family of threat models spanning a wide range of realistic scenarios. Following the threat modeling direction, Martimiano et al. [13] recognize user-constrained devices as significant attack vectors, and suggest a *multi-attacker* approach to better capture diverse, *context-dependent* scenarios, thus integrating device security tightly into human-centered ceremony verification.

In [2], Basin, Radomirović, and Schmidt investigate *human errors* and their impact on the protocol security. They formalize different degrees of *human fallibility* and, using existing authentication protocols, show that security guarantees depend on the chosen user profile. Bella, Giustolisi, and Schürmann study *indirect human threats* against technical system, e.g., via interactions with other humans with no intent to collude [4]. In [14], Sempreboni and Viganò identify typical *mutations in human behavior* and analyze their impact on the ceremony execution. Results obtained in all these works have been automated using (extensions of) Tamarin [1] which reviled some efficiency limitations in using protocol verification tools for ceremony analysis.

The ceremony modeling framework developed by Fila and Radomirović in [10] builds on the rich body of human- and device-centered modeling approaches described above. It unifies the representation of humans, devices, network components, and other entities, while tracking agents' possessions, knowledge, ownership, control, states, and more. The objective of our investigation is to further advance the ceremony modeling process, by scrutinizing the CM framework with a special focus on dishonest, erroneous, faulty, or adversarial behavior.

3 Formal framework for ceremony modeling and analysis

To keep this paper self-contained, we summarize in this section the main building blocks of the CM framework developed in [10], and explain how to design a model that we call *idealized*, and which expresses the intended flow of a ceremony.

Throughout this paper, we use a toy ceremony, referred to as *the STM ceremony*, illustrated on the left in Figure 1. This generic ceremony allows two humans to interact over a channel. It can be used to communicate cryptographic messages over a network, modeling a conversation between people over a voice channel, or showing emotions (e.g., happiness or anger) using a visual channel.

3.1 Typed algebra for ceremony specification

In the rest of this paper, $\mathcal{C}$ stands for a generic ceremony. In order to specify ceremony $\mathcal{C}$, the CM framework uses a typed signature

$$\Sigma_{\mathcal{C}} = ((\texttt{Types}_{\mathcal{C}}, \preceq_{\mathcal{C}}), \mathbb{V}_{\mathcal{C}}, \mathbb{C}_{\mathcal{C}}, \mathbb{F}_{\mathcal{C}}),$$

where $(\texttt{Types}_{\mathcal{C}}, \preceq_{\mathcal{C}})$ is a partially ordered set of user-defined types that are written using `this font`, $\mathbb{V}_{\mathcal{C}}$ is a set of typed variables, $\mathbb{C}_{\mathcal{C}}$ is a set of typed constants, and $\mathbb{F}_{\mathcal{C}}$ is a set of typed function symbols of arity greater than 0.

Types allow us to distinguish resources of different nature. A set of types useful for the STM ceremony is given on the right in Figure 1. Partial order $\preceq_{\mathcal{C}}$ enables statements such as "voice is a special kind of channel", formally `voice` $\preceq_{\mathcal{C}}$ `chan`, i.e., `voice` is a subtype of `chan`. Subtype relation $\preceq_{\mathcal{C}}$ is reflexive.

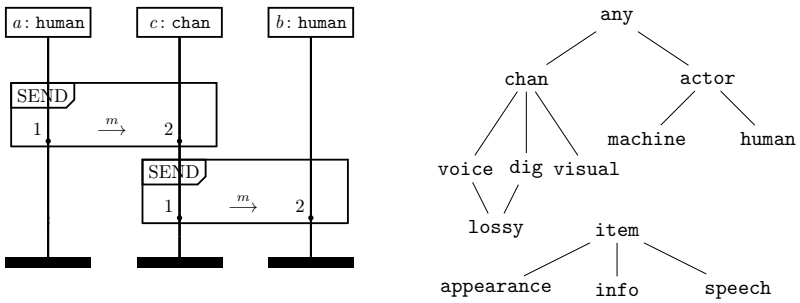

Fig. 1: Toy STM ceremony and the underlying, partially ordered type structure

Given $x \in \mathbb{V}_{\mathcal{C}}$ (resp. $C \in \mathbb{C}_{\mathcal{C}}$), we use $x\colon \mathtt{t}$ (resp. $C\colon \mathtt{t}$) to declare the type of variable x (resp. of constant C). Typed function symbols are declared by specifying the types of the function's inputs and output, as follows $f\colon \mathtt{t}_1, \ldots, \mathtt{t}_n \to \mathtt{t}$.

Entities taking part in a ceremony are called *agents*, and items of any type (messages, physical objects, locations, etc.) being manipulated by the agents are called *resources*. In ceremonies, the difference between agents and resources is fuzzy. For instance, when the cellphone sends a message, it acts as an agent, but when it is passed between two humans it is a resource. Agents can thus be seen as special kinds of resources. Formally, resources (including agents) are modeled using typed terms over signature $\Sigma_{\mathcal{C}}$. The set of all terms (resp. ground terms, i.e., terms without variables) over $\Sigma_{\mathcal{C}}$ is denoted by $\mathbb{T}(\Sigma_{\mathcal{C}})$ (resp. $\mathbb{GT}(\Sigma_{\mathcal{C}})$).

In order to link a ceremony specification to its execution, the CM framework uses substitutions. Recall that a *substitution* σ is a mapping from variables to terms $\sigma\colon \mathbb{V}_{\mathcal{C}} \to \mathbb{T}(\Sigma_{\mathcal{C}})$, such that $\sigma(x) = x$ for all but finitely many $x \in \mathbb{V}_{\mathcal{C}}$. The set $Dom(\sigma) = \{x \in \mathbb{V}_{\mathcal{C}} \mid \sigma(x) \neq x\}$ is called the *domain* of σ. Substitution $\sigma\colon \mathbb{V}_{\mathcal{C}} \to \mathbb{T}(\Sigma_{\mathcal{C}})$ can be recursively extended to all terms from $\mathbb{T}(\Sigma_{\mathcal{C}})$, by setting $\sigma(f(u_1, \ldots, u_n)) = f(\sigma(u_1), \ldots, \sigma(u_n))$, for any $f \in \mathbb{F}_{\mathcal{C}}$ and any $u_1, \ldots, u_n \in$

$\mathbb{T}(\Sigma_{\mathcal{C}})$. We say that substitution σ' *extends* substitution σ (written $\sigma \subseteq \sigma'$) if $\sigma'(x) = \sigma(x)$, for every $x \in Dom(\sigma)$. All substitutions in the CM framework must be *well-typed*, i.e., type of $\sigma(x)$ must be a subtype of the type of x.

3.2 Availability relation

A core building block of the CM framework is a binary *availability relation* AV, which models agents' possessions, in a broad sense, e.g., information knowledge, legal or physical ownership, physical containment, perception of feelings, etc.

In standard protocol models, possession is reduced to knowledge of cryptographic information and is assumed monotonously increasing. In the world of ceremonies, availability relation is *not monotonous*. For instance, humans forget passwords or learn new secrets, they lose, give away or find physical objects, they change location, their mood can vary, they can sell or buy a property, etc.

3.3 Transformations

When agents engage in a ceremony, their possessions evolve. To model this fact, the CM framework uses transformations. A *transformation* describes an atomic step of a ceremony: it specifies the type and the number of participants, as well as the resources they must possess before and after the transformation is executed. Transformations may be specified using tables, as in Figure 2, or with an inline

T	Pre	Post
p_1	$\mathbf{d}_1$	$\mathbf{d'}_1$
$\vdots$	$\vdots$	$\vdots$
p_n	$\mathbf{d}_n$	$\mathbf{d'}_n$

SEND	Pre	Post
x: `any`	z : `item`	z : `item`
y: `any`		z : `item`

LOSE	Pre	Post
x: `any`	z : `item`	

Fig. 2: A generic T, SEND and LOSE transformation specifications

notation where columns are separated by "|" and rows by ";". The inline notation for a generic transformation T from Figure 2 is $T[p_1|\mathbf{d}_1|\mathbf{d'}_1; \ldots; p_n|\mathbf{d}_n|\mathbf{d'}_n]$, where

- T is the transformation's *name*,
- n is the transformation's *arity*,
- $p_1, \ldots, p_n$ denote the transformation's *participants*,
- $\mathbf{d_i}$ is a tuple encoding the transformation's *preconditions*, i.e., resources that participant p_i must posses so that the transformation can be executed,
- $\mathbf{d'_i}$ is a tuple encoding the transformation's *postconditions*, i.e., resources that participant p_i will acquire after the transformations has been executed.

Upon execution of transformation T, each participant p_i loses the availability of resources represented by $\mathbf{d_i}$ and gains the availability of those represented by $\mathbf{d'_i}$.

Transformations are symbolic schemes, parameterized by typed variables. When transformations are used in a ceremony specification or execution, these variables are instantiated with the help of well-typed substitutions.

3.4 Ceremony specification

In a nutshell, ceremony specification acts as a blueprint: it defines entities, called *roles*, involved in the ceremony, and lists the steps the agents playing these roles must perform. Each such step is called an *action*, and corresponds to an instantiated row of a transformation. More formally, an action is a single-line table $\mathrm{T}_i[r|\mathbf{v}|\mathbf{v}']$, where $\mathrm{T}[p_1|\mathbf{d}_1|\mathbf{d}'_1;\ldots;p_n|\mathbf{d}_n|\mathbf{d}'_n]$ is a transformation, $i \in \{1,\ldots,n\}$, and r (resp. $\mathbf{v}$ and $\mathbf{v}'$) are terms (resp. tuples of terms) satisfying $\tau(p_i) = r$, $\tau(\mathbf{d}_i) = \mathbf{v}$, *and* $\tau(\mathbf{d}'_i) = \mathbf{v}'$ for some well-typed substitution τ. Actions inherit the Pre and Post columns from the corresponding transformations.

Following [10], we formally define a ceremony as follows.

Definition 1. *A ceremony $\mathcal{C}$ is a tuple $\mathcal{C} = (\Sigma_\mathcal{C}, \mathcal{T}_\mathcal{C}, \mathcal{R}_\mathcal{C}, Spec_\mathcal{C})$, where $\Sigma_\mathcal{C} = ((\mathtt{Types}_\mathcal{C}, \preceq_\mathcal{C}), \mathbb{V}_\mathcal{C}, \mathbb{C}_\mathcal{C}, \mathbb{F}_\mathcal{C})$ is a typed signature, $\mathcal{T}_\mathcal{C}$ is a set of transformations over $\Sigma_\mathcal{C}$, $\mathcal{R}_\mathcal{C} \subseteq \mathbb{T}(\Sigma_\mathcal{C})$ is a set of* roles *involved in $\mathcal{C}$, and $Spec_\mathcal{C}$ is a function mapping ceremony roles to sequences of actions. Given a role $r \in \mathcal{R}_\mathcal{C}$, the sequence $Spec_\mathcal{C}(r)$ is called specification of role r in $\mathcal{C}$.*

For technical reasons, we require that the set of variables appearing in the domain and image of $Spec_\mathcal{C}$ is disjoint from the set of variables appearing in the definitions of transformations from $\mathcal{T}_\mathcal{C}$.

Example 1. Our toy STM ceremony illustrated in Figure 1 can be formalized as $\mathcal{STM} = (\Sigma, \{\mathrm{SEND}\}, \{a, b\colon \mathtt{human}, c\colon \mathtt{chan}, \}, Spec)$, where a partially ordered typed structure underlying Σ has been given in Figure 1, SEND is the transformation specified in the middle of Figure 2, $m\colon \mathtt{item}$ represents a resource to be transmitted, and the three roles specifications are:

$$Spec(a) = \mathrm{SEND}_1[a|m|m]$$
$$Spec(c) = \mathrm{SEND}_2[c|\ \ |m] \cdot \mathrm{SEND}_1[c|m|m]$$
$$Spec(b) = \mathrm{SEND}_2[b|\ \ |m].$$

Remark 1. A role specification represents a sequence of actions that captures the perception and behavior of a single agent executing that role. These actions are connected through shared variables, which enables references to previously obtained resources and encodes resources flow from the point of view of the role.

3.5 Ceremony execution

When ceremonies are executed, actions in roles specifications are instantiated using well-typed substitutions. Formally, ceremony executions are modeled using *traces*. A trace is a list of events representing steps that have been executed by the agents. Two types of events may occur in a ceremony trace:

- RUN CREATION event (RC) modeling the activation of a run for a role,
- TRANSFORMATION event representing the execution of a transformation by a number of agents.

When a run is activated, an agent and potentially some specific resources are linked with one of the ceremony roles r and its specification. This link is modeled using a substitution that will be extended throughout the ceremony execution. Formally, run is a triple $(\theta, \sigma^\theta, Spec_{\mathcal{C}}(r))$, where θ is a run identifier, σ^θ is a substitution such that $\sigma^\theta(r) \in \mathbb{GT}(\Sigma_{\mathcal{C}})$, and $Spec_{\mathcal{C}}(r)$ is the sequence of actions that agent $\sigma^\theta(r)$ must follow to participate in the execution of $\mathcal{C}$.

An agent can take part in the execution of transformation T only if: its run has already been activated, the first upcoming action to be executed according to this run corresponds to transformation T, and the agent possess (in terms of the availability relation AV) all resources specified by the Pre column of that action. Existence of n active runs in which T_i is the name of the first upcoming action, for $i \in \{1, \ldots, n\}$ is thus a necessary condition for execution of an n-ary transformation T. After a successful transformation execution, the participating *runs evolve*: executed actions are removed from the specifications and the runs' substitutions are *extended* by instantiating relevant variables with the resources gained by the agents during the transformation execution. Furthermore, the availability relation is updated accordingly, to reflect on resources gained or lost by the agents that have executed the transformation.

Remark 2. The following key points characterize the ceremony execution:

1. A transformation event in a trace is obtained by combining instantiated actions of agents playing ceremony roles and participating in the transformation execution (one action per role). *Every transformation event must thus correspond to a well-typed instantiated transformation table.*
2. Every transformation event present in a trace entails evolution of the runs that participated in the transformation execution. This implies that *actions from a role specification are executed only once and in the order given by the specification sequence.*
3. To capture parallel executions, *several runs* – to be executed by the same or different agents – *may be activated for a role, but they have to use distinct identifiers.*

Example 2. Consider *Alice* who has proven that $P = NP$ – which obviously has groundbreaking consequences for cryptography foundations – and submits paper $P13$: `info` describing this result to the STM workshop. To do so, she sends her work to the STM chair *Bob* via submission server S: `dig`, accordingly to the STM ceremony specified in Example 1. *Alice* has already written her paper, which we model with the initial availability relation $AV = \{(Alice, P13)\}$. If all agents behave honestly, the expected submission process can be modeled with the following trace

E_1. $\mathrm{RC}(1,\ \sigma^1 = \{a \mapsto Alice, m \mapsto P13\},\ Spec(a) = \mathrm{SEND}_1[a|m|m])$

E_2. $\mathrm{RC}(2,\ \sigma^2 = \{c \mapsto S\}, Spec(c) = \mathrm{SEND}_2[c|\ |m] \cdot \mathrm{SEND}_1[c|m|m])$

E_3. $\mathrm{RC}(3,\ \sigma^3 = \{b \mapsto Bob\}, Spec(b) = \mathrm{SEND}_2[b|\ |m])$

E_4.

SEND	Pre	Post
Alice	*P*13	*P*13
S		*P*13

E_5.

SEND	Pre	Post
S	*P*13	*P*13
Bob		*P*13

At the end of the submission process, the availability relation becomes $AV = \{(Alice, P13), (S, P13), (Bob, P13)\}$ modeling that *Bob* got *Alice*'s submission.

Remark 3. A substitution in a run must be consistent with the current availability relation so that an action can be executed. However, the objectives of these two elements are different. A substitution in a run keeps the *history* of all resources that have been used so far by the corresponding agent, while the availability relation models *current possessions* of agents. This implies that once the substitution in a run assigns a value to a variable, this assignment is permanent and will never be forgotten, while the availability set of an agent can both increase and decrease as a result of the ceremony execution.

3.6 Context

The success or failure of a ceremony execution often depends on external conditions or environmental factors. To capture such dependencies, the CM framework embeds a ceremony in a supporting context, representing prerequisite or auxiliary interactions. Formally speaking, *context* is a set of ceremonies that can be executed in parallel to the studied ceremony (that we call *primary* ceremony in what follows). In the presence of a non-empty context, the same agents may participate in the primary and in (some of) the contextual ceremonies. An execution trace may thus mix events coming from any of these ceremonies. It is important to notice that, at a given point in time, every agent has only one availability set that can be updated by the primary or contextual ceremony events.

Example 3. Consider *Alice* who meets the STM chair *Bob* in person and questions him about the acceptance of her paper, using again the STM ceremony. She thus sends message $P13?$ (encoding her question) to *Bob* over voice channel V. The result of her request will depend on the context in which the ceremony is executed. In a quiet room, *Bob* will hear her question impeccably, whereas in a packed concert hall the request may go unnoticed. To model the second situation, we accompany the STM ceremony by the context containing *losing* ceremony $\mathcal{L} = (\Sigma_{\mathcal{L}}, \text{LOSE}, \{l\}, Spec_{\mathcal{L}})$, where the LOSE transformation is specified on the right of Figure 2, role $l\colon \texttt{lossy} \preceq_{\mathcal{L}} \texttt{chan}$ represents a lossy channel, with $Spec_{\mathcal{L}}(l) = \text{LOSE}_1[l|m\colon \texttt{info}|\ ]$. The corresponding execution trace

E_1. RC(1, $\sigma^1 = \{a \mapsto Alice, m \mapsto P13?\}$, $Spec(a) = \text{SEND}_1[a|m|m]$)
E_2. RC(2, $\sigma^2 = \{c \mapsto V\}, Spec(c) = \text{SEND}_2[c|\ |m] \cdot \text{SEND}_1[c|m|m]$)
E_3. RC(3, $\sigma^3 = \{l \mapsto V\}, Spec_{\mathcal{L}}(l) = \text{LOSE}_1[l|m|\ ]$)

E_4.

SEND	Pre	Post
Alice	$P13?$	$P13?$
V		$P13?$

E_5.

LOSE	Pre	Post
V	$P13?$	

models that *Bob* did not get *Alice*'s question and thus will be unable to answer.

4 Practical guidelines for ceremony specification

So far, we have explained how to come up with an *idealized specification* of a ceremony, i.e., a specification that is tailored to an expected execution where

all agents follow their roles' specifications faithfully, as in Example 1 and 2. In real life, however, agents may be malicious or erroneous, they may cheat, omit or insert ceremony steps. The idealized specification cannot cover this kind of unforeseen behavior, and is therefore unpractical from the point of view of the ceremony analysis and assessment. In this section, we design heuristics for practical ceremony specification, allowing the modeler to devise models capturing the intended flow of a ceremony as well as its unexpected variants.

4.1 Modeling (in)secure and (un)reliable channels

The authors of the CM framework notice in [10] that, "due to the atomicity of the transformations, any channel involved in a ceremony must be modeled explicitly as a role participating in the ceremony". To ensure a desired flow of resources in a ceremony, actions present in a role specification usually share common variables, as explained in Remark 1. Such consistency between actions is crucial for expected executions as it enforces data integrity: once a variable is bound by a substitution, its value is retained throughout the run. Nevertheless, the variable reuse in a role specification harms the model's flexibility.

Limitation 1 *The dependency constraint introduced by variable reuse across several actions in a role specification prevents the representation of agents that are dishonest, reactive, or careless, and selectively modify data between actions.*

As it is the case in the idealized specification of our STM ceremony from Example 2, regular channel role specification uses the same variable to denote the resource to be received and to be sent in the next ceremony step. However, such a design choice disables traces where a malicious channel modifies received resource before transmitting it further.

To address the limitation introduced by variables reuse in a role specification, we propose to decouple the actions of a role by making them independent.

Modeling strategy 1 (*Role splitting*) *To accommodate agents able to intentionally or inadvertently alter the manipulated resources, split the corresponding role into multiple roles, each with a specification containing exactly one action from the original role specification. By assigning fresh roles, the corresponding runs and substitutions are made independent, and an implicit dependency induced by variable sharing is thus removed.*

Algorithm formalizing the above modeling strategy is presented in Appendix A.

Modeling strategy 1 is particularly well-suited for roles representing Dolev–Yao channels [7] or corrupted devices able of freely modifying or injecting messages.

Example 4. Suppose that the STM submission server S is controlled by a fraudulent owner who, to enrich himself, extorts money from the authors before transferring their papers to the chair *Bob*. If the authors have not paid, the server owner replaces their paper with a scientifically empty submission $\emptyset$ (generated by an artificial intelligence) that will surely be rejected.

The above scenario cannot be modeled using the idealized $\mathcal{STM}$ specification as in Example 2. Upon the execution of event E_4, variable m from the first action of $Spec(c) = \text{SEND}_2[c|\ |m] \cdot \text{SEND}_1[c|m|m]$ would be instantiated with $P13$ and this assignment must remain unchanged for the remaining actions in this run.

To capture the fraudulent server scenario, we set $AV = \{(Alice, P13), (S, \emptyset)\}$, and we modify the idealized specification from Example 1 according to Modeling strategy 1. We split role c into two roles c', c'': `chan` with respective specifications $Spec(c') = \text{SEND}_2[c'|\ |m]$ and $Spec(c'') = \text{SEND}_1[c''|m|m]$. Variables m in specifications of c' and c'' are now local and can therefore be instantiated with different values, leading to the trace illustrating the malicious server behavior:

E_1. RC(1, $\sigma^1 = \{a \mapsto Alice, m \mapsto P13\}$, $Spec(a) = \text{SEND}_1[a|m|m]$)
E_2. RC(2, $\sigma^2 = \{c' \mapsto S\}, Spec(c') = \text{SEND}_2[c'|\ |m]$
E_3. RC(3, $\sigma^3 = \{c'' \mapsto S, m \mapsto \emptyset\}, Spec(c'') = \text{SEND}_1[c''|m|m]$
E_4. RC(4, $\sigma^4 = \{b \mapsto Bob\}, \text{SEND}_2[b|\ |m]$)

E_4.

SEND	Pre	Post
Alice	*P*13	*P*13
S		*P*13

E_5.

SEND	Pre	Post
S	$\emptyset$	$\emptyset$
Bob		$\emptyset$

Remark that Modeling strategy 1 cannot be applied blindly to every channel, but should rather depend on the channel characteristics. Consider *secure* channels, i.e., those ensuring integrity, confidentiality and authentication, and *reliable* channels, i.e., those that always deliver messages or resources.

- Since we have a certainty that the resources sent over *secure and reliable* channel are authentic, cannot be altered, and will always be delivered, there is no need to model these channels as ceremony roles. Instead, the sender can interact directly with the recipient (e.g., *Alice* can give a physical object to *Bob*; *Alice* can talk to *Bob* if they are in a close vicinity, etc.).
- Specification of channels that are *secure but unreliable* can be expressed using sequences of actions, as in the idealized specification. When exchanging over a secure channel, there is a certainty that the resource and its origin will not be modified. Different actions from the channel role specification may thus share variables. Unreliability of the channel will be captured by the trace if the channel agent stops executing its role at a certain point.
- To fully capture the capabilities of *insecure channels*, the role splitting modeling strategy should be applied.

4.2 Transformation design

The CM framework does not come with any pre-defined transformation and the modeler is responsible for defining transformations that correspond to the desired atomic steps of a ceremony under study. A transformation specification describes how information and resources flow between agents. To preserve consistency and model honest behavior, variables are typically shared between the participants in the Pre and Post columns of a transformation. To enforce the transformation integrity, during the ceremony execution, all such shared variables must be instantiated with the same term, as observed in point 1 of Remark 2.

Limitation 2 *Variable reuse in a transformation specification prevents the model from capturing situations where data might change, e.g., due to fault injection, memory errors or human mistakes.*

Consider the SEND transformation as in the idealized $\mathcal{STM}$ specification, which uses variable z across its both rows. When a SEND event will appear in a trace, all occurrences of z will have to be instantiated with the same term. A trace based on such idealized specification cannot thus capture infrastructural faults, e.g., a bit flip during a transmission, that would result in the network receiving a different resource from the one sent by the agent playing role a.

To address Limitation 2, we propose to relax the forced equality of terms that can instantiate multiple occurrences of a variable in a transformation.

Modeling strategy 2 (*Relaxing transformation specification*) *To accommodate modeling of faulty infrastructure or human erroneous behavior, relax the specification of a transformation they may participate in by replacing (some of the) multiple occurrences of the same variable with distinct, fresh variables. The following steps guide the transformation specification relaxation:*

1. *Locate the variable whose integrity should be loosened – typically the datum that might be corrupted, mistyped, etc.*
2. *Replace each occurrence of the variable that needs to not equal the original one with a distinct, fresh variable of the same type.*

Let us study the process of distinguishing variables on an example.

Example 5. Possible versions of the SEND transformation resulting from the Modeling strategy 2 application could correspond to the following situations:

a)

SEND	Pre	Post
x	z	z
y		z'

Transmission fault

b)

SEND	Pre	Post
x	z	z'
y		z

Sender memory mistake

c)

SEND	Pre	Post
x	z	z'
y		z'

Spoofing or typo

To keep the model robust and meaningful, we recommend to only distinguish variables that lead to modeling realistic scenarios, as illustrated in Example 5. Application of Modeling strategy 2 should always be justified by analyzing which types of agents could introduce errors or adversarial behavior. Furthermore, to allow for an efficient analysis of the resulting model, this strategy should be applied in a controlled way and to selected transformations only. Widespread relaxation must be avoided unless modeling systemic faults, because over-generalization augments the number of possible traces and complicates formal reasoning.

Example 6. Suppose that the author *Alice*, the STM submission server S and the chair *Bob* are honest, but the infrastructure on which the sever is implemented is faulty and may randomly inject faults to the messages passed from the authors to the server. *Alice* submits her paper where she proved that $P = NP$, but due to certain bit flips some occurrences of $=$ become $\neq$, and the submission

platform receives a corrupted version of the article. A valuable contribution of *Alice*'s work is lost and the paper will surely be rejected.

The above scenario cannot be modeled using the original specification of SEND from Figure 2, because $\text{SEND}_1[Alice|P13|P13]$ and $\text{SEND}_2[S|\ \ |P13']$ cannot be combined in a valid instantiation of $\text{SEND}[a|z|z;c|\ \ |z]$.

We apply Modeling strategy 2 and use version a) of SEND from Example 5, allowing server S to receive a paper different from the one sent by *Alice*. With this relaxation, we capture the trace

E_1. $\text{RC}(1,\ \sigma^1 = \{a \mapsto Alice, m \mapsto P13\},\ Spec(a) = \text{SEND}_1[a|m|m])$
E_2. $\text{RC}(2,\ \sigma^2 = \{c \mapsto S\}, Spec(c) = \text{SEND}_2[c|\ \ |m] \cdot \text{SEND}_1[c|m|m])$
E_3. $\text{RC}(3,\ \sigma^3 = \{b \mapsto Bob\}, Spec(b) = \text{SEND}_2[b|\ \ |m])$

E_4.

SEND	Pre	Post
Alice	*P*13	*P*13
S		*P*13′

E_5.

SEND	Pre	Post
S	*P*13′	*P*13′
Bob		*P*13′

where the corrupted version on the paper is provided to *Bob*.

From a theoretical perspective, whenever two variables have distinct names, they represent totally independent resources. However, relaxing a transformation specification as proposed by Modeling strategy 2 is in practice guided by scenarios where the decoupled resources are expected to be different but *close* to each other, according to some domain-specific notion of closeness, e.g., permuted letters in text, flipped bits in bitstrings, some physical item and its counterfeit. To represent such a (weaker than equality) dependency in a symbolic model, we may introduce a special agent, called *Sim* (for *Similarity*), whose objective is to keep track of tolerably close values. We then narrow the Modeling strategy 2 by extending the transformation of interest with a new row representing *Sim*, and including to its Pre and Post columns pair(s) of variables that are supposed to represent close resources. For instance, the SEND transformation representing a transmission fault in Example 5 would become $\text{SEND}[x|z|z;y|\ \ |z';Sim|\langle z,z'\rangle|\langle z,z'\rangle]$. Under this specification, resources that can instantiate variables z and z' are restricted to those that are declared as *close* in the availability set of agent *Sim*. For instance, the availability relation in Example 6 would need to satisfy $(Sim, \langle P13, P13'\rangle) \in AV$.

Making use of the *Sim* agent to grasp a similarity between resources provides a fine-grained control mechanism for Modeling strategy 2. By expressing soft equality, the modeler can capture better real-world error patterns, such as minor typing mistakes, partial data loss, bit flips, etc.

4.3 Context construction

Agents participating in a ceremony must follow a finite sequence of steps. Accordingly, role specifications are modeled as sequences of actions that are executed *exactly once* and in the *given order*. While the strategy summarized by point 2 of Remark 2 is necessary for idealized specifications, it severely restricts the ability to model adversarial or atypical scenarios.

Limitation 3 *Constraints introduced by linearly ordered actions in idealized role specifications and evolution of runs during a ceremony execution makes it impossible to model situations where steps are skipped, replayed, or arbitrarily inserted.*

To lift the above limitation and allow for more flexibility, we propose to augment the primary ceremony with a carefully chosen context. The idea is to make some selected transformations available as standalone units, decoupled from any fixed sequence of actions.

Modeling strategy 3 (*Context design*) *To support actions that may be repeated or occur out of order, identify the underlying transformation and create a context ceremony containing this transformation as well as n roles (where n is the arity of the transformation) with single-action specifications corresponding each to an individual row of the transformation.*

Formally, let $\mathcal{C} = (\Sigma_{\mathcal{C}}, \mathcal{T}_{\mathcal{C}}, R_{\mathcal{C}}, Spec_{\mathcal{C}})$ be the primary ceremony, and let $T[p_1|\mathbf{d}_1|\mathbf{d'}_1; \ldots; p_n|\mathbf{d}_n|\mathbf{d'}_n] \in \mathcal{T}_{\mathcal{C}}$ be the transformation whose actions we wish to make repeatable or executable out of order. Modeling strategy 3 consists of constructing (or augmenting an existing) context $\mathcal{K} = (\Sigma_{\mathcal{K}}, \mathcal{T}_{\mathcal{K}}, \mathcal{R}_{\mathcal{K}}, Spec_{\mathcal{K}})$ by including T to $\mathcal{T}_{\mathcal{K}}$, and adding to $\mathcal{R}_{\mathcal{K}}$ n fresh roles r^i, each having the specification $Spec_{\mathcal{K}}(r_i) = T_i[r_i|\tau_i(\mathbf{d}_i)|\tau_i(\mathbf{d'}_i)]$, where τ_i is some well-typed substitution.

This solution maintains the integrity of the original modeling approach for the primary ceremony (its idealized specification stays untouched), while enabling out-of-band interactions such as replays, message injections, or opportunistic retransmissions. Each of these actions can now be executed repeatedly by creating a new run for the corresponding role.

It is important to ensure that the introduced context remains meaningful, tractable, and faithful to real-world scenarios. Two complementary approaches may guide the decision-making process for selecting relevant transformations:

- *Role based selection:* Identify the roles that might be played by agents capable of replaying, reordering, or injecting actions, e.g., Dolev-Yao channels. Then, include in the context all transformations associated with these roles.
- *Transformation based selection:* Analyze the meaning of each transformation to determine whether it corresponds to a real-life situation that may reasonably happen multiple times or out of order. Actions involving sending of cryptographic messages are typical good candidates: such messages are often cached or remembered by the sender and can easily be resent or replayed, making them suitable for context modeling. In contrast, transformations involving a transfer of physical items or legal rights are hardly repeatable: once the object is given away or the right is transferred, the necessary preconditions to execute the event one more time do no longer hold.

Example 7. Consider a denial-of-service (DOS) scenario where a malicious script sitting at the STM submission platform forwards an important number of randomly generated papers to the STM chair *Bob*, without any actual articles being submitted by real authors. The idealized $\mathcal{STM}$ specification cannot capture the

above situation, because the server cannot execute action SEND_1 (i.e., sending of a resource to agent executing role b) without having executed SEND_2 before (i.e., without having received the resource from agent executing role a).

To solve the problem, we follow Modeling strategy 3, and introduce a context ceremony $\mathcal{D} = (\Sigma, \{\text{SEND}\}, \{snd, rcv\}, Spec_{\mathcal{D}})$, with the roles specifications $Spec_{\mathcal{D}}(snd) = \text{SEND}_1[snd|p|p]$ and $Spec_{\mathcal{D}}(rcv) = \text{SEND}_1[rcv|\ \ |p]$.

By mingling events originating from the primary STM ceremony and the context ceremony $\mathcal{D}$, we obtain a trace modeling the flooding of Bob with k valueless submissions:

L_j. $\text{RC}(j,\ \sigma^j = \{snd \mapsto S, p \mapsto P_j\},\ Spec_{\mathcal{D}}(snd) = \text{SEND}_1[snd|p|p])$, for $j \in \{1, \ldots, k\}$

L_{k+j}. $\text{RC}(k+j,\ \sigma^{k+j} = \{b \mapsto B\},\ Spec(b) = \text{SEND}_2[b|\ \ |m]),$ for $j \in \{1, \ldots, k\}$

L_{2k+j}.

SEND	Pre	Post
S	P_j	P_j
Bob		P_j

, for $j \in \{1, \ldots, k\}$.

To finish this section, we remark that, although the traces enabled by implementing Modeling strategy 1 (*Role splitting*) and Modeling strategy 3 (*Context design*) are similar, the objectives of the two strategies are different:

- Modeling strategy 1 *modifies* the idealized specification of the ceremony under study, and should therefore be used only if *all* agents capable of executing the affected role are subject to potential dishonest behavior.
- Modeling strategy 3 preserves the original specification of the primary ceremony, but allows the modeler to enable, at their discretion, a particular context and study how the primary ceremony behaves under some specific conditions. Finally, several different contextual ceremonies could be defined to supply the analysis of the primary ceremony with a large spectrum of execution possibilities.

We also stress that all our modeling strategies are *conservative*, i.e., they adapt a ceremony specification in such a way that *all* traces possible under the original specification remain valid and can be expressed using the updated models.

5 Case study: two-factor authentication ceremony

Modeling strategies 1–3 have been designed based on our personal experience in manual specification of ceremonies within the CM framework. The toy STM ceremony is however too simplistic to illustrate all possible issues, assess the quality of the proposed measures, and convince the modelers about their utility. We therefore performed a larger case study on which we could validate our findings. Due to space restrictions, we only present its main results and we refer the reader to [11] for its detailed description and analysis.

The ceremony we consider reflects common real-life deployments of a two-factor authentication (2FA) procedure. We selected 2FA for its real-world relevance [9] as well as diversity of involved actors, resources, and interaction channels implying rich attack surface. Our goal was to demonstrate that the CM

framework – once augmented with our modeling strategies – can faithfully express both the *expected* and *faulty* executions of a real-world ceremony.

W started by providing an idealized specification of the 2FA ceremony and exhibited a trace modeling its expected information flow. We also defined a number of security properties crucial to guarantee the objectives of 2FA. We then gradually updated the idealized specification according to Modeling strategies 1–3 and tested whether the desired properties still hold in the updated model.

5.1 Two-factor authentication ceremony

Formal specification of our 2FA ceremony is given in Appendix B. The ceremony is about a user authenticating with an online bank, and develops as follows:

The user who wants to connect to their bank, types the bank's URL, allowing them to access an authentication session and enter login credentials (username and password). The credentials are sent to the bank's server which performs a verification against a user record stored in a bank's database. If the credentials are valid, the system generates a one-time password (OTP) and sends it to the phone number associated with the user's account. The user receives the OTP on their phone and manually inputs it into the authentication session. This OTP is then sent to the bank server for verification. If the OTP is correct, the server confirms authentication by returning a success message to the user.

5.2 Case study results

We have analyzed the 2FA ceremony under the assumptions that humans are fallible and devices may be compromised. We were especially interested in the following security properties formalized using Tamarin-like syntax:

1. Every time when a user, who wants to perform an authentication session with a specific bank, gets an OTP, they must have provided their email address and password to the authentication session with that same bank.

$$\exists u\colon \texttt{human}, w\colon \texttt{intention}, b\colon \texttt{service}, o, e, p\colon \texttt{info}, \#j, \#i, \#l, \text{ s.t.},$$
$$i < l < j \ \wedge \ \big(\text{DISPLAY}_2[u| - |o, -]@j \wedge \text{TYPE}[u|w|-; w|b|-]@i$$
$$\implies \text{INPUT}_1[u|\langle e, p\rangle, \mathit{session}(b)|-]@l\big)$$

2. Whenever the user sees an OTP on their phone, that exact value must have already been sent to their device by the bank in the same execution.

$$\exists o\colon \texttt{info}, c\colon \texttt{device}, n\colon \texttt{number}, \#j, \#i, \text{ s.t., } i < j \wedge$$
$$\big(\text{DISPLAY}_2[-|c|o, c]@j \implies \text{OTP}[-|\langle G, -, o, n\rangle|\langle -, o\rangle; c|n|n, o]@i\big)$$

By updating the idealized specification of the 2FA ceremony with the help of modeling strategies from Section 4, we were able to obtain attack traces – unreachable in the original, idealized model – corresponding to real-life attacks.

Application of Modeling strategy 2 and 3 provided a model capturing an attack resulting from *human error* and violating Property 1: *Alice* wishes to authenticate to her bank, but mistypes the URL and ends up in a session that

corresponds to a phishing site mimicking her bank. Unaware of her error, she submits her email and password to the compromised session. Using a standard man-in-the-middle strategy, the hacker controlling the phishing site then uses *Alice*'s credentials to complete a login procedure with *Alice*'s bank on her behalf.

Application of Modeling strategy 1 gave rise to a specification capturing behavior of a *compromised device*, leading to a DOS attack preventing *Alice* from authenticating, and violating Property 2: The bank sends a legitimate OTP to *Alice*'s compromised phone, which instead displays some forged value. *Alice* enters the forged OTP into the authentication session and her authentication request fails, because the bank has never issued the code she is providing.

Thanks to the 2FA case study, we could put the modeling strategies developed in this work into practice and confirm that the CM framework enjoys expressive power required for realistic ceremony-level analysis. We could study the specification of unusual roles, like human intentions and fresh randomness generators. Finally, by specifying context-dependent resources, like OTP, we could experience the practical pertinence of a non-monotonous availability relation.

6 Conclusion and future work

The main objective of this work was to put the ceremony modeling framework developed in [10] into practice and study its expressiveness, especially from the point of view of modeling unexpected, yet realistic, situations. We have shown that a natural, idealized way of specifying a ceremony captures the intended flow of resources, but imposes restrictions that prevent the representation of faulty, erroneous, or malicious behavior, leaving a broad class of realistic attack traces unreachable. We then proposed three pragmatic modeling strategies to guide the modeler in a construction of a ceremony model close to an idealized specification but powerful enough to capture real-life adversarial scenarios. Our modeling heuristics have been successfully validated on a two-factor authentication case study, grounding the abstract CM framework in a security-critical, real-world scenario, building confidence in the developed targeted modeling strategies, and paving the way toward systematic modeling of other complex ceremonies.

Due to a critical impact of human aspect on cybersecurity, ceremonies will certainly be intensely studied in the upcoming years. The most essential direction to be investigated in the field is the automation of ceremony verification. It has been demonstrated that protocol verification tools, such as Tamarin [1], exhibit performance limitations when faced with the complexity underlying ceremonies, which combines inference modulo equational theories, custom type structures, personalized transformations, and non-monotonous availability relation [10]. By introducing modeling heuristics that both guide the construction of good-quality ceremony models and lend themselves to automation, our work paves the way for developing specialized tools for ceremony verification. A crucial follow-up will be to apply these automated tools to a broader range of case studies – today handled manually and at considerable time cost – to demonstrate the scalability, wide applicability, and the adaptability of our approach.

Acknowledgments. This work received funding from the France 2030 program managed by the French National Research Agency, under grant agreement No. ANR-22-PECY-0006.

References

1. Basin, D.A., Cremers, C., Dreier, J., Meier, S., Sasse, R., Schmidt, B.: Tamarin Prover, https://tamarin-prover.github.io/, accessed on 04.07.2025
2. Basin, D.A., Radomirovic, S., Schmid, L.: Modeling Human Errors in Security Protocols. In: CSF. pp. 325–340. IEEE Computer Society (2016)
3. Bella, G., Coles-Kemp, L.: Layered analysis of security ceremonies. In: SEC. IFIP Advances in Information and Communication Technology, vol. 376, pp. 273–286. Springer (2012)
4. Bella, G., Giustolisi, R., Schürmann, C.: Modelling human threats in security ceremonies. J. Comput. Secur. **30**(3), 411–433 (2022)
5. Carlos, M.C., Martina, J.E., Price, G., Custódio, R.F.: A Proposed Framework for Analysing Security Ceremonies. In: SECRYPT. pp. 440–445. SciTePress (2012)
6. Carlos, M.C., Martina, J.E., Price, G., Custódio, R.F.: An updated threat model for security ceremonies. In: SAC. pp. 1836–1843. ACM (2013)
7. Dolev, D., Yao, A.C.: On the security of public key protocols. IEEE Trans. Inf. Theory **29**(2), 198–207 (1983)
8. Ellison, C.M.: Ceremony design and analysis. IACR Cryptol. ePrint Arch. p. 399 (2007), http://eprint.iacr.org/2007/399
9. EU: Commission Delegated Regulation (EU) 2018/389 of 27 November 2017 (2017)
10. Fila, B., Radomirović, S.: Nothing is Out-of-Band: Formal Modeling of Ceremonies. In: CSF. pp. 464–478. IEEE (2024)
11. Hoxha, E.: Formal Modeling of Realistic Behavior in Security Ceremonies: A Feasibility Study. Master's thesis, Politecnico di Milano, Italy (2025)
12. Internet Crime Complaint Center: 2024 Internet Crime Report. Tech. rep., Federal Bureau of Investigation (2024), https://www.ic3.gov/AnnualReport/Reports/2024_IC3Report.pdf
13. Martimiano, T., Martina, J.E., Olembo, M.M., Carlos, M.C.: Modelling User Devices in Security Ceremonies. In: STAST. pp. 16–23. IEEE CS (2014)
14. Sempreboni, D., Viganò, L.: A mutation-based approach for the formal and automated analysis of security ceremonies. J. Comput. Secur. **31**(4), 293–364 (2023)
15. Verizon Business: 2025 Data Breach Investigations Report. Tech. rep., Verizon Business (2025), https://www.verizon.com/business/resources/T4b0/reports/2025-dbir-data-breach-investigations-report.pdf, accessed on 13.07.2025
16. World Economic Forum: Global Cybersecurity Outlook 2025 (2025), https://www.weforum.org/publications/global-cybersecurity-outlook-2025/, accessed on 12.07.2025

A The role splitting algorithm

Modeling strategy 1 is formalized using Algorithm 1.

Algorithm 1 Role splitting algorithm for Modeling strategy 1

Require: Ceremony $(\Sigma_{\mathcal{C}}, \mathcal{T}_{\mathcal{C}}, \mathcal{R}_{\mathcal{C}}, Spec_{\mathcal{C}})$; $r \in \mathcal{R}_{\mathcal{C}}$, $Spec_{\mathcal{C}}(r) = Act^1 \cdot Act^2 \cdot \ldots \cdot Act^k$, $k > 1$
Ensure: Updated ceremony $(\Sigma_{\mathcal{C}}, \mathcal{T}_{\mathcal{C}}, \mathcal{R}'_{\mathcal{C}}, Spec'_{\mathcal{C}})$
1: $\mathcal{R}'_{\mathcal{C}} \leftarrow \mathcal{R}_{\mathcal{C}} \setminus \{r\}$
2: $Spec'_{\mathcal{C}} \leftarrow Spec_{\mathcal{C}} \setminus \{Spec_{\mathcal{C}}(r)\}$
3: **for** $j = 1$ **to** k **do**
4: Let $r^j \notin \mathcal{R}_{\mathcal{C}}$ be a fresh role name
5: $\mathcal{R}'_{\mathcal{C}} \leftarrow \mathcal{R}'_{\mathcal{C}} \cup \{r^j\}$
6: Let $Act^j = T_i[r|\mathbf{v}_i|\mathbf{v}'_i]$ be the j-th action in $Spec_{\mathcal{C}}(r)$
7: Let ρ^j be a substitution defined by:

$$\rho^j(x) = \begin{cases} r^j & \text{if } x = r \\ x & \text{otherwise} \end{cases}$$

8: $Spec'(r^j) \leftarrow T_i[\rho^j(r)|\rho^j(\mathbf{v}_i)|\rho^j(\mathbf{v}'_i)]$
9: $Spec'_{\mathcal{C}} \leftarrow Spec'_{\mathcal{C}} \cup \{Spec'(r^j)\}$
10: **end for**
11: **return** $(\Sigma_{\mathcal{C}}, \mathcal{T}_{\mathcal{C}}, \mathcal{R}'_{\mathcal{C}}, Spec'_{\mathcal{C}})$

B The 2FA case study specification

A high-level representation of our 2FA ceremony is illustrated in Figure 6. Formally, the 2FA ceremony is specified as a tuple $\mathcal{A} = (\Sigma_{\mathcal{A}}, \mathcal{T}_{\mathcal{A}}, \mathcal{R}_{\mathcal{A}}, Spec_{\mathcal{A}})$:

- It relies on a signature whose type structure is depicted in Figure 3.
- It involves seven roles, $R_{\mathcal{A}} = \{u, w, session(b), b, d, c, Source\}$, representing a user, the user's will to authenticate to their bank, the bank's authentication session, the bank itself, a database storing bank's customers credentials, the user's cellphone, and the *Source* agent responsible for generating fresh values, such as nonces and one-time passwords.
- The set $\mathcal{T}_{\mathcal{A}}$ of transformations used by ceremony $\mathcal{A}$ is given in Figure 4.
- The idealized specifications of roles are given in Figure 5.

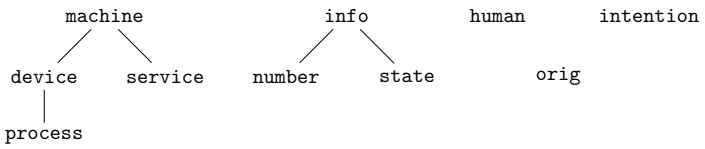

Fig. 3: The hierarchy of types used in the 2FA ceremony model.

TYPE (TE)	Pre	Post
x: human	y	$session(z)$
y: intention	z : service	z

INPUT (IN)	Pre	Post
x: human	z: info, y	z, y
y: process		z

LOGIN (LO)	Pre	Post
x: process	z: info	m
y: server		$\langle z, m\rangle$
Source: orig	m: info	

EXISTS (EX)	Pre	Post
x: server	f, $\langle c_1$: info, c_2: info, m: info$\rangle$	f, $\langle G, m, t, k\rangle$
f: info	$\langle k$: number, $c_1, c_2\rangle$	$\langle k, c_1, c_2\rangle$
Source: orig	t: info	

OTP (OT)	Pre	Post
x: server	$\langle G, m$: info, t: info, k: number$\rangle$	$\langle m, t\rangle$
y: device	k	k, t

DISPLAY (DI)	Pre	Post
x: device	z: info	z
y: human	x	z, x

VALIDATE (VA)	Pre	Post
x: process	m, t: info	m
y: server	$\langle m, t$: info$\rangle$	$\langle G, m\rangle$

ACCEPT (AC)	Pre	Post
x: server	$\langle G, m$: info$\rangle$	$\langle G, m\rangle$
y: process	m: info	OK

Fig. 4: Transformations used by the 2FA ceremony

$Spec_{\mathcal{A}}(u) =$
$\quad \mathrm{TE}_1[u|w|s(b)]\cdot$
$\quad \mathrm{IN}_1[u|\langle e,p\rangle,\ s(b)|\langle e,p\rangle,\ s(b)]\cdot$
$\quad \mathrm{DI}_2[u|c|o,c]\cdot$
$\quad \mathrm{IN}_1[u|o,s(b)|o,s(b)]\cdot$
$\quad \mathrm{DI}_2[u|s(b)|OK,s(b)]$

$Spec_{\mathcal{A}}(b) =$
$\quad \mathrm{LO}_2[b|\ |\langle e,p,id\rangle]\cdot$
$\quad \mathrm{EX}_1[b|d,\langle e,p,id\rangle|d,\langle G,id,o,n\rangle]\cdot$
$\quad \mathrm{OT}_1[b|\langle G,id,o,n\rangle|\langle id,o\rangle]\cdot$
$\quad \mathrm{VA}_2[b|\langle id,o\rangle|\langle G,id\rangle]$
$\quad \mathrm{AC}_1[b|\langle G,id\rangle|\langle G,id\rangle]$

$Spec_{\mathcal{A}}(session(b)) =$
$\quad \mathrm{IN}_2[session(b)|\ |\langle e,p\rangle]\cdot$
$\quad \mathrm{LO}_1[session(b)|\langle e,p\rangle|id]\cdot$
$\quad \mathrm{IN}_2[session(b)|\ |o]\cdot$
$\quad \mathrm{VA}_1[session(b)|id,o|id]\cdot$
$\quad \mathrm{AC}_2[session(b)|id|OK]\cdot$
$\quad \mathrm{DI}_1[session(b)|OK|OK]$

$Spec_{\mathcal{A}}(w) = \mathrm{TE}_2[w|b|b]$
$Spec_{\mathcal{A}}(d) = \mathrm{EX}_2[d|\langle n,e,p\rangle|\langle n,e,p\rangle]$
$Spec_{\mathcal{A}}(c) = \mathrm{OT}_2[c|n|n,ot]\cdot\mathrm{DI}_1[c|ot|ot]$
$Spec_{\mathcal{A}}(Source) =$
$\quad \mathrm{LO}_3[Source|id|\ ]\cdot\mathrm{EX}_3[Source|o|\ ]$

Fig. 5: Idealized specification of roles in the 2FA ceremony

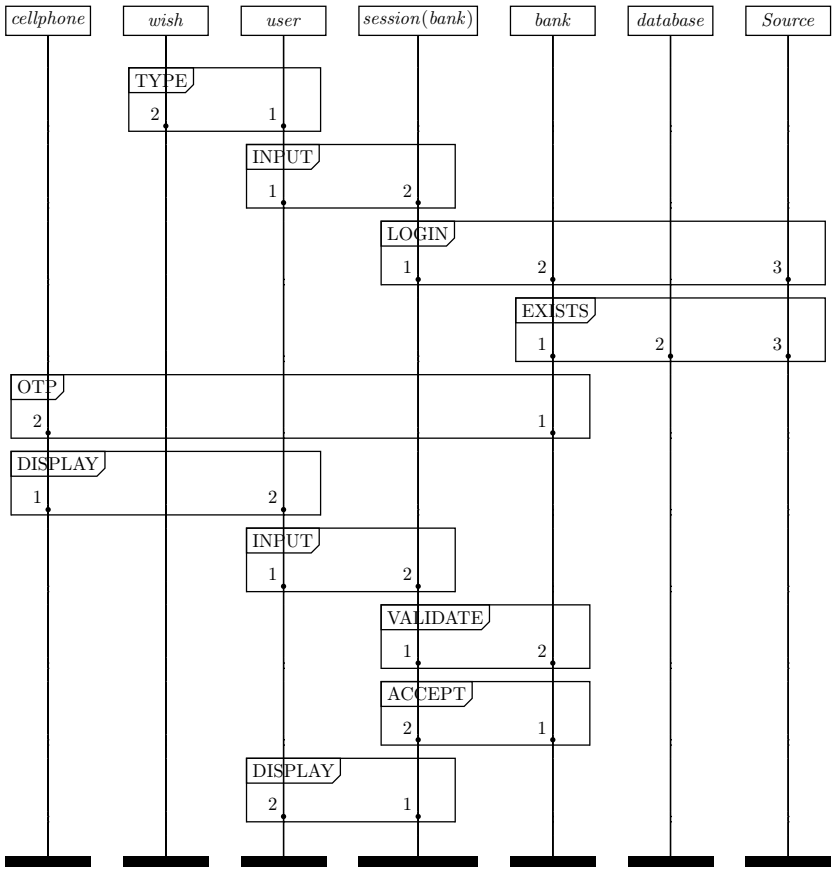

Fig. 6: High-level specification of the 2FA ceremony.