Distributed Temporal Logic for the Analysis of Security Protocol Models

David Basin
Department of Computer Science
ETH Zurich, Switzerland

Carlos Caleiro  Jaime Ramos
SQIG - Instituto de Telecomunicações and Department of Mathematics
IST, TU Lisbon, Portugal

Luca Viganò
Department of Computer Science
University of Verona, Italy

August 25, 2011

Abstract

The distributed temporal logic DTL is an expressive logic, well-suited for formalizing properties of concurrent, communicating agents. We show how DTL can be used as a metalogic to reason about and relate different security-protocol models. This includes reasoning about model simplifications, where models are transformed to have fewer agents or behaviors, and verifying model reductions, where to establish the validity of a property it suffices to consider its satisfaction on only a subset of models.

We illustrate how DTL can be used to formalize security models, protocols, and properties, and then present three concrete examples of metareasoning. First, we prove a general theorem about sufficient conditions for data to remain secret during communication. Second, we prove the equivalence of two models for guaranteeing message-origin authentication. Finally, we relate channel-based and intruder-centric models, showing that it is sufficient to consider models in which the intruder completely controls the network. While some of these results belong to the folklore or have been shown, mutatis mutandis, using other formalisms, DTL provides a uniform means to prove them within the same formalism. It also allows us to clarify subtle aspects of these model transformations that are often neglected or cannot be specified in the first place.

1 Introduction

Security protocols are distributed programs that employ cryptography in order to achieve their objectives in possibly hostile environments. They have been widely studied within the formal methods community as they are difficult to design and notoriously prone to error: it is difficult to predict all the possible ways that distributed computation may proceed and thereby foresee all the ways that an intruder can overcome cryptography by exploiting the willingness of honest agents to communicate. The research in this area is extensive and a number of logics and formalisms exist for specifying and verifying security protocols.

Our contribution in this paper is not another method or tool for protocol analysis. Rather, we present a logical foundation for formalizing and reasoning about models of security protocols, which are at the heart of other methods and tools. The foundation is based on DTL, a distributed temporal logic [34]. DTL is an expressive, general-purpose logic that is well-suited for formalizing
both local, agent-specific properties and global properties of distributed communicating processes. Within DTL, we formalize different theories for security protocol analysis, capturing different ways that agents can interact with the network in the presence of an active intruder. We use these theories to show how DTL can be used in two, quite distinct ways:

- as an object logic for formalizing specific protocol models and proving properties of protocols with respect to these models, and
- as a metalogic for relating different models and proving metatheorems about the models themselves.

To show how DTL can be used as an object logic, we consider the well-known NSPK protocol [54] and its corrected version NSL [42]. We use these to illustrate DTL’s application to both protocol falsification and verification. As mentioned above, there are many existing formalisms for reasoning about protocols. However, unlike existing formalisms for object-level protocol analysis, DTL is additionally suitable as a metalogic, and the possibility of using DTL effectively as an object logic is a necessary prerequisite for more advanced metalogical applications. Conversely, an effective metalogic aids object logic applications in that we can derive general metatheorems useful for protocol verification. We will illustrate this by establishing a general result about sufficient conditions for data to remain secret during communication.

To further show how DTL can be used as a metalogic, we present applications to formalizing and verifying properties of security protocol models and translations between models. Our motivations here are both theoretical and practical. Within the security community, a wide variety of different models have been proposed, often with slightly differing assumptions, concerning communication, the powers of the intruder, and the abstractions used in describing security protocols. Theoretically, we would like to understand, and have a formal foundation for establishing, the relationship between these different models. Practically, when building security protocol analysis tools, we would like to reduce the number of different scenarios that must be searched (model reduction) as well as to simplify the scenarios considered (model simplification). Such optimizations can substantially improve the efficiency of algorithmic verification tools and should be employed whenever possible, provided they preserve the properties of interest. The challenge, of course, is to formalize and prove this. Our contribution is to show how DTL can be used to give a simple, rigorous, and uniform account of different model reduction and simplification techniques as well as other kinds of metatheoretic properties of security protocol models. We do this by studying concrete models, showing how they can be formalized naturally and reasoned about using DTL.

First, we formalize and prove the equivalence of two models for guaranteeing message-origin authentication. The first model is an abstract model where principals may use a special channel, controlled by a trusted third party, that logs all incoming messages and issues evidence of their origin to the recipients. The second model is more concrete and uses digital signatures to implement an authentic channel. We present two different transformations of the corresponding DTL models and corresponding notions of equivalence based on preserving translations of properties. This example is not aimed at justifying a model simplification, but rather at showing how DTL can be used to relate models at different levels of abstraction. The equivalences we prove can be used to justify that concrete (signature-based) designs achieve the more abstract notion of message-origin authentication defined in terms of a trusted third party.

Afterwards, we use DTL as a metalogic to explore the exact meaning of different modeling assumptions that are common in security protocol analysis. Namely, even when working with an intruder who controls the network, one must make numerous modeling decisions. For example, is the intruder identified with the network? That is, does he coexist with the network and intercept messages sent to the network versus is he identified with the network and all messages are sent directly to him? Alternatively, does one consider all possible interleavings between intruder actions and those of honest agents or only some subset? In particular, under suitable assumptions, we establish the following results:

(1) the intruder can be identified with the network;
(2) The steps (actions) of honest agents can be “compressed” in the sense that the receipt of a message can always be immediately followed by the agent sending a response; and

(3) All distribution in the model can be eliminated by considering all actions from the intruder’s point of view.

The results (1) and (3) constitute model simplifications as they involve a translation between models (and also properties), leading to models that are simpler in the sense that they involve fewer agents or collapse behaviors. Result (2) exemplifies a model reduction, where to establish the validity of a property it suffices to consider its satisfaction on only a subset of models, namely just those models where steps are compressed.

While some of the results we prove belong to the folklore or have, mutatis mutandis, already been shown using other formalisms, our logic provides a means to prove them in a general, uniform way within the same formalism. It also allows us to clarify aspects of these properties that are often neglected or cannot be specified in the first place. For example, the equivalences proved in the message-origin example depend critically on the capabilities of the intruder using the trusted channel, what exactly is signed, and that different keys are used for different purposes.

While other logics or formalisms could be used for these tasks, we believe that DTL offers a number of advantages. Security protocols are carried out by distributed agents, with individual state, who concurrently execute protocols, synchronizing over shared communication. DTL provides a rich language with a corresponding semantics that naturally captures all of these aspects. It has a distributed dimension that captures the agents, their local state, and communication. Concurrent execution, as well as the formalization of security properties, is captured by adding a temporal dimension based on past and future time linear temporal operators. The logic is quite flexible and avoids commitment to particular models and properties, which are formalized as DTL theories. The semantics of DTL is based on interpretation structures, which are a model of concurrent, distributed systems that is well-suited for carrying out semantic reasoning about, and transformation of, such systems.

A strength of our DTL formalization is that it allows us to spell out all the fine details of the security proofs that we give in this paper. As is well known, this is particularly important in the area of Information Security, where researchers often have well-developed intuitions, but their models and proofs are prone to subtle errors. Although we have developed a tableau system for DTL [7, 8], our proofs in this paper are semantic. We prefer semantic arguments because they are shorter and far more intuitive than tableau proofs. In contrast, proofs with the tableau system can be machine checked. We note, in this regard, that the validity of DTL formulas can be decided by using a trace-consistent translation to LTL [8], which also makes DTL amenable to model checking.

We proceed as follows. In Section 2 we introduce the distributed temporal logic DTL. In Section 3, we define in DTL a protocol-independent distributed communication model, on top of which protocols and security goals can be formalized and analyzed, as shown in Section 3.3. In Section 4 we illustrate how DTL can be used as an object logic to either verify security protocols or construct counter-examples to their claimed properties. However, the results are established as a corollary of a metalevel result about secrecy that we state and prove. The core results of the paper are in Section 5, where we present the metalevel results, mentioned above. We draw conclusions and discuss related and future work in Section 6.

2 Distributed temporal logic

The distributed temporal logic DTL is a logic for reasoning about temporal properties of distributed systems from the local point of view of the system’s agents, which are assumed to execute sequentially and to interact by synchronous event sharing. In this paper, we use DTL theories given, in lato sensu, by classes of models satisfying axioms and other defining conditions. Note that the set of axioms may be infinite, resulting in validity being undecidable in the theory, despite the decidability result mentioned above. This is the case for the theories that we later present.
2.1 Syntax

The logic is defined over a distributed signature

$$\Sigma = \langle \text{Id}, \{\text{Act}_i\}_{i \in \text{Id}}, \{\text{Prop}_i\}_{i \in \text{Id}} \rangle,$$

where \(\text{Id}\) is a finite set of agent identifiers and, for each \(i \in \text{Id}\), \(\text{Act}_i\) is a set of local action symbols and \(\text{Prop}_i\) is a set of local state propositions. In a nutshell, the actions \(\text{Act}_i\) correspond to true statements about an agent when they have just occurred and the state propositions \(\text{Prop}_i\) characterize the current local states of the agents. Following standard protocol terminology, we will also refer to the agents participating in a protocol execution as principals.

The global language \(\mathcal{L}\) is defined by the grammar

$$\mathcal{L} ::= @_{i_1}[\mathcal{L}_{i_1}] | \cdots | @_{i_n}[\mathcal{L}_{i_n}] | \bot | \mathcal{L} \Rightarrow \mathcal{L},$$

for \(\text{Id} = \{i_1, \ldots, i_n\}\), where the local languages \(\mathcal{L}_i\) for each \(i \in \text{Id}\) are defined by

$$\mathcal{L}_i ::= \text{Act}_i \mid \text{Prop}_i \mid \bot \mid \mathcal{L}_i \Rightarrow \mathcal{L}_i \mid \mathcal{L}_i \cup \mathcal{L}_i \mid \mathcal{L}_i \cap \mathcal{L}_i \mid \mathcal{L}_i \cap \mathcal{L}_i \mid \mathcal{L}_i \cap ^\circ \mathcal{L}_j,$$

with \(j \in \text{Id}\). As notation, we will use \(\gamma\) and \(\delta\) for global formulas, and \(\phi\) and \(\psi\) for local formulas.

A global formula \(\theta_i[\phi]\) means that the local formula \(\phi\) holds for agent \(i\). Local formulas, as the name suggests, hold locally for the different agents. For instance, locally for an agent \(i\), the operators \(U\) and \(S\) are the usual (strong) until and since temporal operators, while the communication formula \(\bigcirc_j[\psi]\) means that agent \(i\) has just communicated (synchronized) with agent \(j\), for whom \(\psi\) holds.\(^1\)

As notation, we write \(\mathcal{L}_i^\circ\) to denote the set of all purely temporal formulas of \(\mathcal{L}_i\), that is, excluding communication formulas. We call \(\phi \in \mathcal{L}_i^\circ\) a private formula. Furthermore, if \(\phi\) does not contain the temporal operators \(U\) and \(S\), then we call it a state formula. Finally, we write \(\mathcal{L}^\circ\) to denote the set of all global formulas built from private formulas.

Other logical connectives (conjunction, disjunction, etc.) and temporal operators can be defined as abbreviations, for example:

- \(X \phi \equiv \bot \cup \phi\) tomorrow (next)
- \(F \phi \equiv \top \cup \phi\) sometime in the future
- \(F_0 \phi \equiv \phi \lor F \phi\) now or sometime in the future
- \(G \phi \equiv \neg F \neg \phi\) always in the future
- \(G_0 \phi \equiv \phi \land G \phi\) now and always in the future
- \(\phi W \psi \equiv (G \phi) \lor (\phi U \psi)\) weak until (unless)
- \(Y \phi \equiv \bot \lor S \phi\) yesterday (previous)
- \(P \phi \equiv \top \lor S \phi\) sometime in the past
- \(P_0 \phi \equiv \phi \lor P \phi\) now or sometime in the past
- \(H \phi \equiv \neg P \neg \phi\) always in the past
- \(H_0 \phi \equiv \phi \land H \phi\) now and always in the past
- \(\phi B \psi \equiv (H \phi) \lor (\phi S \psi)\) weak since (back to)
- \(* \equiv H \bot\) in the beginning
- \(\phi \gg_j \psi \equiv \phi \Rightarrow \bigcirc_j[\psi]\) calling

Here we use the subscript \(\circ\) to denote the reflexive versions of operators. Note also that calling is specific to DTL as it involves communication: \(\theta_i[\phi \gg_j \psi]\) means that if \(\phi\) holds for agent \(i\) then he calls (synchronizes with) agent \(j\), for whom \(\psi\) must hold.

2.2 Semantics

The protocol models that we consider are based on partially-ordered sets of events with labeling information. We employ sequences (of events or labels) to represent protocol executions and we

\(^1\)Note that the DTL syntax here differs slightly from the original presentation in [34]. Previously, the operator \(\bigcirc_i\) was overloaded with \(\theta_i\) and its interpretation was therefore context dependent.
Figure 1: A distributed life-cycle for agents A, B, and C.

\[
\pi_A(\emptyset) \xrightarrow{a_A(e_1)} \pi_A(\{e_1\}) \xrightarrow{a_A(e_4)} \pi_A(\{e_1, e_4\}) \xrightarrow{a_A(e_5)} \pi_A(\{e_1, e_4, e_5\}) \xrightarrow{a_A(e_8)} \ldots
\]

Figure 2: The progress of agent A.

write \(w = (w_1, w_2, w_3, \ldots)\) to denote a possibly infinite sequence whose elements are \(w_1, w_2, w_3, \ldots\). Furthermore, we write \(|w|\) to denote the length of the sequence \(w\), where \(|\emptyset| = 0\) for the empty sequence \(\emptyset\) and \(|w| = \infty\) whenever \(w\) is infinite. Finally, we write \(w.w'\) to denote the concatenation of two sequences, provided that the first sequence is finite, and we write \(w_i\) to denote the prefix of \(w\) of length \(i\), i.e. \(w_i = (w_1, \ldots, w_i)\), provided that \(0 \leq i \leq |w|\). Clearly, \(w_{|0|} = \emptyset\).

The interpretation structures of \(\mathcal{L}\) are labeled distributed life-cycles, built upon a simplified form of Winskel’s event structures [66] (see also [67] for the relationship to other concurrency models). A local life-cycle of an agent \(i \in \text{Id}\) is a countable (finite or infinite), discrete, and well-founded total order \(\lambda = (\mathcal{E}_i, \leq_i)\), where \(\mathcal{E}_i\) is the set of local events and \(\leq_i\) the local order of causality. We define the corresponding local successor relation \(\rightarrow_i \subseteq \mathcal{E}_i \times \mathcal{E}_i\) to be the relation such that \(e \rightarrow_i e'\) if \(e <_i e'\) and there is no other \(e''\) such that \(e <_i e'' <_i e'\). As a consequence, \(\leq_i = \rightarrow_i^*\), i.e., \(\leq_i\) is the reflexive, transitive closure of \(\rightarrow_i\).

A distributed life-cycle is a family \(\lambda = \{\lambda_i\}_{i \in \text{Id}}\) of local life-cycles such that \(\leq = (\bigcup_{i \in \text{Id}} \leq_i)^*\) defines a partial order of global causality on the set of all events \(\mathcal{E} = \bigcup_{i \in \text{Id}} \mathcal{E}_i\). Note that communication is modeled by event sharing, and thus for some event \(e\) we may have \(e \in \mathcal{E}_i \cap \mathcal{E}_j\), with \(i \neq j\). In that case, requiring \(\leq\) to be a partial order amounts to requiring that the local orders are globally compatible, thus excluding the existence of another \(e' \in \mathcal{E}_i \cap \mathcal{E}_j\) such that \(e <_i e'\) but \(e' <_j e\).

We can check the progress of an agent by collecting all the local events that have occurred up to a given point. This yields the notion of the local state of agent \(i\), which is a finite set \(\xi_i \subseteq \mathcal{E}_i\) down-closed for local causality, i.e., if \(e \leq_i e'\) and \(e' \in \xi_i\) then also \(e \in \xi_i\). The set \(\Xi_i\) of all local states of an agent \(i\) is totally ordered by inclusion and has \(\emptyset\) as the minimal element.

In general, each non-empty local state \(\xi_i\) is reached, by the occurrence of an event that we call \(\text{last}(\xi_i)\), from the local state \(\xi_i \setminus \{\text{last}(\xi_i)\}\).\(^2\) The local states of each agent are totally ordered, as a consequence of the total order on local events. Since they are discrete and well-founded, we can enumerate them as follows: \(\emptyset\) is the 0th state; \(\{e\}\), where \(e\) is the minimum of \(\{\mathcal{E}_i, \leq_i\}\), is the 1st state; and if \(\xi_i\) is the \(k\)th state of agent \(i\) and \(\text{last}(\xi_i) \rightarrow_i e\), then \(\xi_i \cup \{e\}\) is agent \(i\)’s \((k + 1)\)th state. We will denote by \(\xi_i^k\) the \(k\)th state of agent \(i\), so \(\xi_i^0 = \emptyset\) is the initial state and \(\xi_i^k\) is the state reached from the initial state after the occurrence of the first \(k\) events. In fact, \(\xi_i^k\) is the only state of agent \(i\) that contains \(k\) elements, i.e., where \(|\xi_i^k| = k\). Given \(e \in \mathcal{E}_i\), \((e \downarrow i) = \{e' \in \mathcal{E}_i \mid e' \leq_i e\}\) is always a local state. Moreover, if \(\xi_i\) is non-empty, then \((\text{last}(\xi_i) \downarrow i) = \xi_i\). Furthermore, for every local state \(\xi_i \neq \mathcal{E}_i\) there exists a unique next event \(\text{next}(\xi_i)\), corresponding to the minimum event in \(\mathcal{E}_i \setminus \xi_i\), such that \(\xi_i \cup \{\text{next}(\xi_i)\}\) is a local state.

We can also define the notion of a global state: a finite set \(\xi \subseteq \mathcal{E}\) closed for global causality, i.e. if \(e \leq e'\) and \(e' \in \xi\), then also \(e \in \xi\). The set \(\Xi\) of all global states constitutes a lattice under inclusion and has \(\emptyset\) as the minimal element. Clearly, every global state \(\xi\) includes the

\(^2\)This statement is only sensible with respect to a given distributed life-cycle. Similar comments also hold for other notions considered below. However, to ease readability, we omit explicit reference to these dependencies whenever they are clear from the context.
local state $\xi_i = \xi \cap Ev_i$ of each agent $i$. Note that we are overloading the notation $\cdot |_i$, which we previously used to denote sequence prefixing, and below we will often write $\xi_i$ for $\xi |_i$. Given $e \in Ev$, $e | e' \in Ev \mid e' \leq e$ is always a global state.

An interpretation structure $\mu = (\lambda, \alpha, \pi)$ consists of a distributed life-cycle $\lambda$ and a family $\alpha = \{\alpha_i\}_{i \in Id}$ and $\pi = \{\pi_i\}_{i \in Id}$ of local labeling functions, where, for each $i \in Id$,

- $\alpha_i : Ev_i \rightarrow Act_i$ associates a local action to each local event, and
- $\pi_i : \Xi \rightarrow \wp(Prop_i)$ associates a set of local state propositions to each local state.

We denote the tuple $(\lambda, \alpha_i, \pi_i)$ also by $\mu_i$.

Fig. 1 depicts a distributed life-cycle, where each row comprises the local life-cycle of one agent. In particular, $Ev_A = \{e_1, e_4, e_5, e_6, \ldots\}$ and $\rightarrow_A$ corresponds to the arrows in $A$’s row. We can think of the occurrence of the event $e_1$ as leading agent $A$ from its initial state $\emptyset$ to the state $\{e_1\}$, and then of the occurrence of the event $e_4$ as leading to state $\{e_1, e_4\}$, and so on. The state-transition sequence of agent $A$ is displayed in Fig. 2. Shared events at communication points are highlighted by the dotted vertical lines. Note that the numbers annotating the events are there only for convenience since, in general, no global total order on events is imposed. Fig. 3 shows the corresponding lattice of global states.

We can then define the global satisfaction relation by

- $\mu, \xi \models \gamma$ if $\mu, \xi, \xi_i \models \gamma$ for every $\xi \in \Xi$,

where the global satisfaction relation at a global state is defined by

- $\mu, \xi, \xi \models \bot$;
- $\mu, \xi, \xi_i \models \gamma \Rightarrow \delta$ if $\mu, \xi, \xi \models \gamma$ or $\mu, \xi_i \models \delta$;
- $\mu, \xi, \xi \models \diamond_i[\varphi]$ if $\mu_i, \xi_i \models \varphi$;

and where the local satisfaction relations at local states are defined by

- $\mu_i, \xi_i \models \act$ if $\xi_i \neq \emptyset$ and $\alpha_i(\text{last}(\xi_i)) = \act$;
- $\mu_i, \xi_i \models p$ if $p \in \pi_i(\xi_i)$;
- $\mu_i, \xi_i \models \bot$;
- $\mu_i, \xi_i \models \varphi \Rightarrow \psi$ if $\mu_i, \xi_i \models \varphi$ or $\mu_i, \xi_i \models \psi$;
- $\mu_i, \xi_i \models \varphi \cup \psi$ if $|\xi_i| = k$ and there exists $\xi^n_i \in \Xi_i$ such that $k < n$ with $\mu_i, \xi^n_i \models \psi$, and $\mu_i, \xi^n_i \models \varphi$ for every $k < m < n$;
- $\mu_i, \xi_i \models \varphi \cup \psi$ if $|\xi_i| = k$ and there exists $\xi^n_i \in \Xi_i$ such that $n < k$ with $\mu_i, \xi^n_i \models \psi$, and $\mu_i, \xi^n_i \models \varphi$ for every $n < m < k$;
- $\mu_i, \xi_i \models \emptyset[\varphi]$ if $|\xi_i| > 0$, last($\xi_i$) $\in Ev_j$, and $\mu_j, (\text{last}(\xi_i) \cup j) \models \varphi$.
If $\mathcal{M}$ is a set of interpretation structures, then we write $\mathcal{M} \models \varphi$ when $\mu \models \varphi$ for all $\mu \in \mathcal{M}$. We will also use $\mathcal{M}_\Xi$ to denote the set of all pairs $(\mu, \xi)$ such that $\mu \in \mathcal{M}$ and $\xi$ is a global state of the distributed life-cycle underlying $\mu$. We say that $\mu$ is a model of $\gamma$ if $\mu$ globally satisfies each of the formulas in $\Gamma$. We say that $\Gamma$ entails $\gamma$ in $\mathcal{L}$ if every global model of $\Gamma$ is also a model of $\gamma$. Given $\Phi \cup \{\psi\} \subseteq \mathcal{L}_i$, we will write $\Phi \models_i \psi$ to denote that every local model of $\Phi$ is also a model of $\psi$, or equivalently, that $\{\@_i[\varphi] \mid \varphi \in \Phi\} \models \@_i[\psi]$.

For instance, the formula $\@_i[p \Rightarrow F \bigcirc_j X q]$ holds in a model if whenever the proposition $p$ holds locally at a state of agent $i$ then there must be a future state of agent $i$ where he has just synchronized with agent $j$, for whom $q$ will hold in the next state.

Fig. 4 illustrates the satisfaction relation with respect to communication formulas of our running example. Clearly $\mu, \emptyset \models B[\psi] \cup \bigcirc_A[\varphi]$, because $\mu, \xi \models B[\bigcirc_A[\varphi]]$. Note however that $\mu, \xi \not\models @_B[\bigcirc_A[\varphi]]$, although $\mu, \xi \models @_A[\varphi]$.

We now establish general rules for reasoning about different classes of formulas.

**Lemma 2.1 (Local properties)** Let $\varphi \in \mathcal{L}_i$ be a local formula and $\mu$ an interpretation structure. Let $\xi, \xi' \in \Xi$ be such that $\xi_i = \xi'_i$. Then $\mu, \xi \models @_i[\varphi]$ if and only if $\mu, \xi' \models @_i[\varphi]$.

**Proof:** Straightforward, using the definition of global satisfaction and the assumption $\xi_i = \xi'_i$. Namely, $\mu, \xi \models @_i[\varphi]$ iff $\mu_i, \xi_i \models_i \varphi$ iff $\mu_i, \xi'_i \models_i \varphi$ iff $\mu, \xi' \models @_i[\varphi]$. \qed

Note that when specifying a distributed system by specifying the local properties of each agent, it makes sense to use local formulas that additionally do not have nested communication. This is because at the specification level, it is not reasonable to require that an agent in a distributed system can be aware of the communication between other agents. As shown in [34], every local DTL formula, even with nested communication, can be expressed by a finite set of local formulas without nested communication.

We now establish a result for local formulas without communication.

**Lemma 2.2 (Private properties)** Let $\varphi \in \mathcal{L}_{\mathcal{O}}^\mathcal{P}$ be a private formula and $\mu$ and $\mu'$ interpretation structures with $\mu_i = \mu'_i$. Let $\xi, \xi' \in \Xi$ be such that $\xi_i = \xi'_i$. Then $\mu, \xi \models @_i[\varphi]$ if and only if $\mu', \xi' \models @_i[\varphi]$.

**Proof:** We prove that $\mu, \xi_i \models_i \varphi$ if and only if $\mu'_i, \xi'_i \models_i \varphi$, by induction on $\varphi$. If $\varphi$ is act then $\mu, \xi_i \models_i \text{act}$ iff $\xi_i \neq \emptyset$ and $\alpha_i(\text{last}(\xi_i)) = \text{act}$ iff $\mu'_i, \xi'_i \models_i \text{act}$. The last equivalence follows from $\mu_i = \mu'_i$ and $\xi_i = \xi'_i$, $\xi'_i \neq \emptyset$, and $\alpha_i(\text{last}(\xi'_i)) = \text{act}$. The case of $p \in \text{Prop}$ is similar and the remaining cases follow by the induction hypothesis. Note that $\varphi \in \mathcal{L}_{\mathcal{O}}^\mathcal{P}$ and thus we do not consider communication formulas. The main result follows then as expected: $\mu, \xi \models @_i[\varphi]$ iff $\mu_i, \xi_i \models_i \varphi$ iff (from what was just proved) $\mu'_i, \xi'_i \models_i \varphi$ iff $\mu', \xi' \models @_i[\varphi]$. \qed

We also have the following invariance rule for global properties.
Proposition 2.3 (Global invariance rule) Let \( \gamma \in \mathcal{L} \) be a global formula, \( \mu \) an interpretation structure, and \( \xi \in \Xi \) a global state. Suppose that (1) \( \mu, \xi \models \gamma \) and (2) \( \mu, \xi' \models \gamma \) for every \( \xi' \in \Xi \) and \( e \in \text{Ev} \setminus \xi' \) such that \( \xi \subseteq \xi' \) and \( \xi' \cup \{e\} \in \Xi \). Then, \( \mu, \xi' \models \gamma \), for every \( \xi' \in \Xi \) such that \( \xi \subseteq \xi' \).

Proof: Let \( \xi' \) be in \( \Xi \) such that \( \xi \subseteq \xi' \). The proof follows by induction on \( \xi' \). If \( |\xi'| = |\xi| \) then \( \xi' = \xi \) and the result follows from the first assumption. Assume now that \( |\xi'| > |\xi| \). Then, \( \xi' = \xi'' \cup \{e\} \) for some \( e \in \text{Ev} \setminus \xi'' \). By the induction hypothesis, \( \mu, \xi'' \models \gamma \) and so, using the second assumption, it also follows that \( \mu, \xi'' \cup \{e\} \models \gamma \). \( \square \)

For local state properties, the invariance rule can be stated in the following more familiar way. Its proof is similar to the proof of Proposition 2.3, mutatis mutandis, and we thus omit it.

Proposition 2.4 (Local invariance rule) Let \( \varphi \in \mathcal{L}_i \) be a local state formula, \( \mu \) an interpretation structure, and \( \xi_i \in \Xi_i \) a local state. Suppose that (1) \( \mu_i, \xi_i \models \varphi \) and (2) \( \mu_i, \xi'_i \models \varphi \) implies \( \mu_i, \xi'_i \cup \{\text{next}(\xi'_i)\} \models \varphi \) for every \( \xi'_i \in \Xi_i \) such that \( \xi_i \subseteq \xi'_i \subseteq \text{Ev}_i \). Then, \( \mu_i, \xi'_i \models \varphi \), for every \( \xi'_i \in \Xi_i \) such that \( \xi_i \subseteq \xi'_i \), or equivalently, \( \mu_i, \xi_i \models \text{G}_0 \varphi \).

Hence, \( \mu \) is a model of \( \mathcal{L}_1[\varphi] \) if and only if \( \mu \) is a model of both \( \mathcal{L}_1[\mathcal{X} \Rightarrow \varphi] \) and \( \mathcal{L}_1[(\varphi \land X_1) \Rightarrow X \varphi] \), or equivalently, \( \mathcal{L}_1[(\varphi \land X \act) \Rightarrow X \varphi] \) for every \( \act \in \text{Act}_1 \).

3 Network and protocol modeling

As we remarked above, DTL supports formal specification and reasoning about models of agents communicating in distributed systems. In this paper, we focus on security protocols where principals interact by exchanging messages through an insecure public channel in an open network. We will specify this network, and protocols on top of it, by defining DTL theories over suitable signatures, which correspond to classes of models.

Security protocols describe how principals exchange messages, built using cryptographic primitives, in order to obtain security guarantees. Our presentation is independent of both the specific algebra of messages considered for the different security protocols and the actions that principals can take during protocol execution. We will thus take both the algebra of messages and the actions as parameters of our models, considering standard examples for concreteness.

More generally, protocol specifications are parametric in the sense that they prescribe a general recipe for communication that can be used by different principals playing in the different protocols roles (sender, responder, server, etc.). The messages transmitted are bit-strings, but, for our purposes, they can also be taken from any other appropriate set of values and our results are independent of such details. We just assume fixed a network signature.

Definition 3.1 A network signature is a pair \((\text{Princ}, \text{Num})\), where \(\text{Princ}\) is a finite set of principal identifiers and \(\text{Num} = \text{Nonces} \uplus \text{SymK} \uplus \text{PubK}\) is a set of “number” symbols used to model atomic data. \(\text{Num}\) is the union of three disjoint sets: \(\text{Nonces}\) is a set of nonce symbols, \(\text{SymK}\) is a set of symmetric key symbols, and \(\text{PubK}\) is a set of public key symbols.

We will use upper-case letters like \(A, B, C, \ldots\), possibly annotated with subscripts and superscripts, to denote principals, \(N\) to denote nonces, and \(K\) to denote shared or public keys.

3.1 Messages

The algebra of messages tells us how messages are constructed. Numerous algebras have been considered in the literature on security protocol analysis, e.g. [26, 50], ranging from the free algebra to various formalizations of algebraic properties of the cryptographic operators employed. The following is a standard example of a free algebra of messages. We will use \(K^{-1}\) to denote the private key that is the inverse of a public key \(K \in \text{PubK}\), and set \(\text{PrivK} = \{K^{-1} \mid K \in \text{PubK}\}\).
**Definition 3.2** Messages, which we denote by \( M \), possibly with annotations, are built inductively from atomic messages (identifiers and number symbols) and private keys, by pairing, encryption, and hashing. For \( M_1 \) and \( M_2 \) messages, we write

- the pairing of \( M_1 \) and \( M_2 \) as \( M_1; M_2 \),
- the symmetric encryption of \( M_1 \) by \( M_2 \) as \( \{ M_1 \}_K^a \),
- the asymmetric encryption of \( M_1 \) by \( K \in PubK \) (respectively, by \( K^{-1} \in PrivK \)) as \( \{ M_1 \}_K^a \) (respectively, \( \{ M_1 \}_K^{-1} \)), and
- the application of a hash function \( H \) to \( M_1 \) as \( H(M_1) \).

We write \( Msg \) to denote the set of messages. \( \triangle \)

Whenever the distinction between symmetric and asymmetric encryption is unimportant, we simply write \( \{ M_1 \}_K \). As usual, we call \( M_2 \) the key and say that \( M_1 \) is in the scope of the encryption; similarly, for \( H(M) \), we say that \( M \) is in the scope of the hash function \( H \). Note that we will often annotate keys with principal names and we will use \( K \) both to denote the public part of the private key \( K^{-1} \) or a symmetric key. Note also that we assume that from a private key \( K^{-1} \) one can compute its public part \( K \) (see, e.g., [61]), which is equivalent to the notion of public-private key pairs that is often also considered. For this reason, we do not consider the inverse of a private key, namely \( (K^{-1})^{-1} \). Observe also that, as we will enforce below, the only ways for a principal to obtain the inverse of a key are to initially know it, to receive it in a message, or when it is a private key that he freshly generated.

As is often done in symbolic approaches to security protocol analysis, we follow the perfect cryptography assumption, which postulates that the cryptographic primitives themselves cannot be attacked and hence the only way to decrypt a message is to possess the appropriate key. We can then define, as is standard, the sets of messages that principals can analyze (decompose) and synthesize (compose), where we have two analysis rules for asymmetric encryption: one for decrypting with a private key \( K^{-1} \) a message \( M \) that has been asymmetrically encrypted with the corresponding public key \( K \), and one for decrypting with the public key \( K \) a message \( M \) that has been asymmetrically encrypted with a private key \( K^{-1} \) (as usual, this corresponds to verifying with key \( K \) a message that has been signed with key \( K^{-1} \)).

**Definition 3.3** We write \( close(S) \) to denote the closure of a set \( S \) of messages under the rules:

\[
\begin{align*}
\frac{M_1; M_2}{M_1} & A_{proj}^a \\
\frac{\{ M \}_K^a}{K^{-1}} & A_{pub} \\
\frac{M}{H(M)} & S_{hash} \\
\frac{M \ K}{\{ M \}_K^a} & S_{symm} \\
\frac{M \ K}{\{ M \}_K^{-1}^a} & S_{pub} \\
\frac{M \ K}{\{ M \}_K^{-1}^a} & S_{priv}
\end{align*}
\]

With the exception of \( A_{privpub} \), all rules are standard and follow the naming convention that rules that decompose messages are labeled with an \( A \) for “analysis” and rules that compose messages with an \( S \) for “synthesis”. For instance, \( A_{pub} \) and \( A_{priv} \) formalize the decryption of a message encrypted with a public key \( K \) and the signature verification of a message signed with a private key \( K^{-1} \). The rule \( A_{privpub} \) is both an analysis and a synthesis rule, as it formalizes the notion of a “publication function” in asymmetric cryptographic systems: given a private key \( K^{-1} \), a publication function computes the corresponding public key \( K \).

Based on these closure rules, we give the following definitions of the content and of the immediate parts of a message, which essentially invert the analysis rules and the synthesis rules, respectively. We then prove some useful properties about sets of secure messages.
Definition 3.4 The content of a message $M$ is the set $\text{cont}(M)$ of messages defined inductively by

$$\text{cont}(M) = \begin{cases} 
\{M\} & \text{if } M \in \text{Num}, \text{ or } M = H(M_1) \text{ for some } M_1, \\
\{K^{-1}\} \cup \text{cont}(K) & \text{if } M = K^{-1} \in \text{PrivK}, \\
\{M\} \cup \text{cont}(M_1) \cup \text{cont}(M_2) & \text{if } M = M_1; M_2, \\
\{M\} \cup \text{cont}(M_1) & \text{if } M = \{[M_1]\}_K.
\end{cases}$$

When $M' \in \text{cont}(M)$, we will often say that $M$ contains $M'$ or that $M'$ is contained in $M$. \hfill \triangle$

Definition 3.5 The immediate parts of a message $M$ is the set $\text{parts}(M)$ of messages defined by

$$\text{parts}(M) = \begin{cases} 
\emptyset & \text{if } M = N \in \text{Nonce} \text{ or } M = K \in \text{SymK} \text{ or } M = K^{-1} \in \text{PrivK}, \\
\{K^{-1}\} & \text{if } M = K \in \text{PubK}, \\
\{M_1, M_2\} & \text{if } M = M_1; M_2, \\
\{M_1, K\} & \text{if } M = \{[M_1]\}_K, \\
\{M_1\} & \text{if } M = H(M_1).
\end{cases}$$

Let $S \subseteq \text{Msg}$ be a set of secret messages, i.e. messages that should not be disclosed. We can then define $S$-secure messages as follows.

Definition 3.6 For $S \subseteq \text{Msg}$, an $S$-secure encryption is either a symmetric encryption $\{[M]\}_K^s$ with $K \in S$ or an asymmetric encryption $\{[M]\}_K^a$ with $K^{-1} \in S$. A message $M$ is said to be $S$-secure if each occurrence of an element of $S$ in the content of $M$ appears either under the scope of an $S$-secure encryption or under the scope of hashing. A message is $S$-insecure if it is not $S$-secure. We write $S$-Sec to denote the set of all $S$-secure messages. \hfill \triangle

The set $S$-Sec messages contains precisely all the messages that can be safely exchanged over an insecure network without revealing the secrets in $S$. If a message $M$ is $S$-insecure, then some element of $S$ must appear in the content of $M$ outside the scope of any $S$-secure encryption or hashing. In that case, we say that the element occurs in the clear (with respect to $S$). We can then prove the two following properties.

Proposition 3.7 If $S \subseteq \text{Msg}$ then $S$-Sec$\cap S = \emptyset$.

This is straightforward: if $M \in S$ then $M$ is not $S$-secure because $M$ itself occurs in the clear.

For reasonable sets of $S$ of secrets, we expect that $S$-Sec is closed. For instance, it would not make sense to require $M_1; M_2$ to be a secret if one allowed both $M_1$ and $M_2$ to be disclosed.

Definition 3.8 We call a set (of secrets) $S \subseteq \text{Msg}$ rational if whenever $S$ contains a message $M$ and $\text{parts}(M) \neq \emptyset$ then $\text{parts}(M) \cap S \neq \emptyset$. \hfill \triangle

This covers the case when $S$ consists only of atomic data and private keys.

Proposition 3.9 For every rational set $S$, we have that close($S$-Sec) = $S$-Sec.

Proof: Clearly, it suffices to prove that close($S$-Sec) $\subseteq$ $S$-Sec. The proof builds on Proposition 3.7 and proceeds by induction on the closure rules. Many of the cases are simple and do not even rely on the rationality of the set $S$. Below are three of the more interesting cases.

$S_{\text{pair}}$: If $M_1; M_2 \notin S$-Sec, then some element of $S$ must occur in the clear in the content of $M_1; M_2$. Then either it occurs in the clear in the content of $M_1$ and therefore $M_1 \notin S$-Sec, or it occurs in the clear in the content of $M_2$ and therefore $M_2 \notin S$-Sec, or else $M_1; M_2$ is itself in $S$. Since $S$ is a rational set, $M_1$ or $M_2$ must also be in $S$, and thus not in $S$-Sec.

$S_{\text{pub}}$: If $\{[M]\}_K^a \notin S$-Sec, then some element of $S$ must occur in the clear in the content of $\{[M]\}_K^a$. Then either it occurs in the clear in the content of $M$ (and the asymmetric encryption is
not $S$-secure, in which case $K^{-1} \not\in S$ and thus $M \not\in S \text{-Sec}$, or $\{M\}^a_{K^{-1}}$ is itself in $S$. Since $S$ is a rational set, $M$ or $K^{-1}$ must also be in $S$, and thus not in $S \text{-Sec}$. Since $S \text{priv}$: If $\{M\}^a_{K^{-1}} \not\in S \text{-Sec}$, then some element of $S$ must occur in the clear in the content of $\{M\}^a_{K^{-1}}$. Then either it occurs in the clear in the content of $M$ and thus $M \not\in S \text{-Sec}$, or $\{M\}^a_{K^{-1}}$ is itself in $S$. Since $S$ is a rational set, $M$ or $K^{-1}$ must also be in $S$, and thus not in $S \text{-Sec}$. □

3.2 A channel-based model

A channel-based signature $\Sigma_{CB}$ is a distributed signature obtained from a network signature $(Princ, Num)$ by taking $Id = Princ \uplus \{Ch\}$, where $Ch$ is the communication channel (used to model asynchronous communication), and defining the action symbols and state propositions of each agent. As an example, consider the following signature of a principal $A$, whose actions $Act_A$ are

- $send(M, B)$: sending the message $M$ to $B$,
- $rec(M)$: receiving the message $M$,
- $spy(M)$: eavesdropping the message $M$, and
- $fresh(X)$: generating a fresh $X \in Nonces \uplus SymK \uplus PrivK$.

$A$’s state propositions $Prop_A$ are

- $knows(M)$: $A$ knows the message $M$.

For the channel, $Ch$, we do not require state propositions, i.e. $Prop_{Ch} = \emptyset$, whereas the actions $Act_{CB}$ include

- $in(A, M, B)$: the message $M$, sent by $A$, arrives on the channel, addressed to $B$,
- $out(A, M, B)$: the message $M$, sent by $A$, is delivered from the channel to $B$, and
- $leak$: leaking of a message.

These actions reflect that the underlying network may be hostile: sending actions name the intended recipient but receiving actions do not name the message’s sender. In fact, we assume, as is standard, that a principal may behave as a Dolev-Yao intruder [33] who can compose, send, and intercept messages at will, but, following the perfect cryptography assumption, cannot break cryptography. Our results, however, are independent of the particular intruder capabilities. We use $\mathcal{L}_{CB}$ to denote the DTL language over the channel-based signature $\Sigma_{CB}$.

The network model we consider here suffices to abstractly formalize and reason about the properties of communication between principals executing security protocols, as well as about protocol models. This model could, of course, be extended in many ways. For example, we could include additional message constructors, additional actions and state propositions, or variants of the ones given, or we could even include servers and additional channels with distinct accessibility and reliability properties (see Section 5 for some examples).

In the channel-based network model $CB$ that we define, principals can send and receive messages at will, always through the channel. If the principal $A$ sends a message to $B$, then the message synchronously arrives at the channel, where it is stored for future delivery to $B$. If delivery ever happens, it must be synchronized with the corresponding receive action of $B$. However, the principal $A$ can only send $M$ to $B$ if $A$ knows both the name $B$ and how to produce the message $M$. As usual, the knowledge of principals is not static. In addition to their initial knowledge, principals gain knowledge from the messages they receive and the fresh data they generate (nonces, symmetric keys, and private keys). Principals may also spy on messages leaked by the channel and learn their content. We do not allow principals to explicitly divert messages, but we also do not guarantee that messages delivered to the channel are ever received.
To ensure that principals only learn new information from the messages they receive and the fresh data they generate, we require that the knows proposition only holds where necessary. We restrict attention to those interpretation structures $\mu$ such that, for every principal $A$, the following condition holds for all messages $M$ and non-empty local states $\xi_A$:

$$(K) \mu, \xi_A \models_A \text{knows}(M) \iff M \in \text{close}(\{M' \mid \mu, \xi_A \models_A (Y \text{knows}(M')) \lor \text{rec}(M') \lor \text{spy}(M') \lor \text{fresh}(M')\})$$

$(K)$ implies that, in every model $\mu = \langle \lambda, \alpha, \pi \rangle$ of the specification, $\pi$ is completely determined by $\lambda$ and $\alpha$, given $\pi_A(\emptyset)$ for each $A \in \text{Princ}$. This is equivalent to saying that the knowledge of each principal only depends on its initial knowledge and on the actions that have occurred. A number of other useful properties follow from $(K)$, e.g., for each principal $A \in \text{Princ}$:

$$\begin{align*}
@_A[\text{knows}(M_1; M_2) \leftrightarrow (\text{knows}(M_1) \land \text{knows}(M_2))] & \quad (K1) \\
@_A[(\text{knows}(M) \land \text{knows}(K)) \Rightarrow \text{knows}(\{M\}K)] & \quad (K2) \\
@_A[(\text{knows}(\{M\}^*_K) \land \text{knows}(K)) \Rightarrow \text{knows}(M)] & \quad (K3.1) \\
@_A[(\text{knows}(\{M\}^*_K) \land \text{knows}(K^{-1})) \Rightarrow \text{knows}(M)] & \quad (K3.2) \\
@_A[\text{knows}(M) \Rightarrow G_k \text{knows}(M)] & \quad (K4) \\
@_A[\text{rec}(M) \Rightarrow \text{knows}(M)] & \quad (K5) \\
@_A[\text{spy}(M) \Rightarrow \text{knows}(M)] & \quad (K6) \\
@_A[\text{fresh}(X) \Rightarrow \text{knows}(X)] & \quad (K7)
\end{align*}$$

To guarantee the freshness and uniqueness of the data generated by each principal, we also require the following axioms, where $M$ ranges over all messages such that $\text{cont}(X) \cap \text{cont}(M) \neq \emptyset$.

$$(F1) \quad @_A[\text{fresh}(X) \Rightarrow Y \neg \text{knows}(M)]$$

$$(F2) \quad @_A[\text{fresh}(X)] \Rightarrow \bigwedge_{B \in \text{Princ} \setminus \{A\}} \bigcirc_B [\neg \text{knows}(M)]$$

Together with $(K7)$, $(F1)$ and $(F2)$ guarantee that every fresh data item is generated at most once, if at all, in each model, and always freshly (also taking into account agents’ initial knowledge). The specification of the network model also contains axioms that characterize the behavior of the channel $Ch$ and of each principal $A \in \text{Princ}$.

$$(C1) \quad @_{Ch}[\text{in}(A, M, B) \Rightarrow_A \text{send}(M, B)]$$

$$(C2) \quad @_{Ch}[\text{out}(A, M, B) \Rightarrow P \text{in}(A, M, B)]$$

$$(C3) \quad @_{Ch}[\text{out}(A, M, B) \Rightarrow_B \text{rec}(M)]$$

$$(C4) \quad @_{Ch}[\text{leak} \Rightarrow (\bigvee_{B \in \text{Princ}} \bigcirc_B [\top])]$$

$$(P1) \quad @_A[\text{send}(M, B) \Rightarrow Y (\text{knows}(M) \land \text{knows}(B))]$$

$$(P2) \quad @_A[\text{send}(M, B) \Rightarrow_{Ch} \text{in}(A, M, B)]$$

$$(P3) \quad @_A[\text{rec}(M) \Rightarrow_{Ch} (\bigvee_{C \in \text{Princ}} \text{out}(C, M, A))]$$

$$(P4) \quad @_A[\text{spy}(M) \Rightarrow_{Ch} (\text{leak} \land P \bigvee_{B,C \in \text{Princ}} \text{in}(B, M, C))]$$

$$(P5) \quad @_A[\bigwedge_{B \in \text{Princ} \setminus \{A\}} \neg \bigcirc_B [\top]]$$

$$(P6) \quad @_A[\text{fresh}(X) \Rightarrow \neg \bigcirc_{Ch}[\top]]$$
The channel axioms (C1)–(C3) are straightforward. They state that a message addressed to A only arrives at the channel if it is sent to A by some principal B; the channel only delivers a message to A if the message for A previously arrived; and if the channel delivers a message to A, then A receives it. (C4) states that when the channel is leaking, some principal is listening.

The principal axioms are also simple. (P1) states a precondition for sending a message: the sender must know both the message and the recipient beforehand. (P2)–(P3) are interaction axioms. (P2) and (P3) state that the sending and receiving of messages must be shared with the corresponding arrival and delivery actions of the channel. (P4) guarantees that a spied message must have arrived at the channel, addressed to some recipient. The last two axioms limit the possible interactions: (P5) guarantees that principals never communicate directly (only through the channel) and (P6) states that actions that generate fresh data are not communication actions.

As our aim is to provide a foundation for modeling security protocols, we will further add, for simplicity, a number of standard restrictions. To start with, we assume there exists a special intruder, also known as the intruder. As is well-known, it suffices to consider one Dolev-Yao intruder, instead of several ones. This can be formally proved using DTL, which we have done in [16] by showing that one intruder is enough (along the lines of the “two (honest) agents are sufficient” result of [24]). We define the set of honest principals to be \( \text{Hon} = \text{Princ} \setminus \{Z\} \). To make sense of the terminology, we must of course ensure that honest principals do not act dishonestly, namely by spying messages. Thus, for every \( A \in \text{Hon} \), we require also that:

\[
(\text{Hon}) \quad @A[\neg \text{spy}(M)], \text{ for every message } M.
\]

We further assume that \( Z \) does not send a message to principal \( A \) if \( A \) will not receive it, or if that same message has already been sent to \( A \). Namely, we assume the following axioms:

\[
(\text{EZ1}) \quad @Z[\text{send}(M, A) \Rightarrow @CH[\text{out}(Z, M, A)]] \text{ and }
\]

\[
(\text{EZ2}) \quad @Z[\text{send}(M, A) \Rightarrow @CH[\neg P \bigvee_{B \in \text{Princ}} \text{in}(B, M, A)]].
\]

This represents a common simplification, which can be made without loss of generality. As shown, for instance in [16], such assumptions do not compromise the model with respect to its ability to represent security-sensitive behaviors, as we will introduce below.

The \( CB \) models \( \mu \) will be those interpretation structures over \( \Sigma_{CB} \) satisfying all these properties.

### 3.3 Modeling security protocols

In this section, we show how to model protocols and properties on top of our channel-based network model. In a typical situation, we will assume a network signature where each principal \( A \in \text{Princ} \) is assigned a private key denoted by \( K_A^{-1} \), whose corresponding public key is the atom \( K_A \). Whereas it is possible (even desirable) that other principals know \( A \)'s public key, we will assume that, at least initially, \( K_A^{-1} \) is known only by \( A \). We formalize this as follows:

\[
(\text{aKey1}) \quad @A[* \Rightarrow \text{knows}(K_A^{-1})]
\]

\[
(\text{aKey2}) \quad @B[* \Rightarrow \neg \text{knows}(M)], \text{ for every } B \in \text{Princ} \setminus \{A\} \text{ and every } M \text{ containing } K_A^{-1}
\]

Similarly, we may assume that there exist symmetric shared atomic keys \( K_{AB} \) for each pair of principals \( A, B \in \text{Princ} \). As above, we require that \( K_{AB} \) is initially known only by \( A \) and \( B \):

\[
(\text{sKey1.1}) \quad @A[* \Rightarrow \text{knows}(K_{AB})]
\]

\[
(\text{sKey1.2}) \quad @B[* \Rightarrow \text{knows}(K_{AB})]
\]

\[
(\text{sKey2}) \quad @C[* \Rightarrow \neg \text{knows}(M)], \text{ for every } C \in \text{Princ} \setminus \{A, B\} \text{ and every } M \text{ containing } K_{AB}
\]

\footnote{Given this distinction between the intruder and the honest participants, we could rewrite several of the axioms by distinguishing the nature of the principals involved, but we refrain from doing so for brevity.}
We may also assume, for simplicity, that all principals \( A, B \in \text{Princ} \) know each other’s names and public keys from the very beginning.

\((\text{N})~ \@_A[* \Rightarrow \text{knows}(B)]\)

\((\text{PK})~ \@_A[* \Rightarrow \text{knows}(K_B)]\)

Such properties may influence the executability of the protocol by the participants, as we have also discussed in [18]. Here, we will simply assume that the initial knowledge of the principals guarantees that the protocols can be executed.

There are several approaches to extracting formal protocol specifications from a protocol description in Alice-and-Bob-style notation, i.e., as a sequence of message exchange steps. Rather than going through the general case, we will illustrate the method by modeling the standard example of the (flawed) simplified Needham-Schroeder Public Key Protocol NSPK [42]. The formalization steps we take are straightforward and they would not be difficult to generate automatically from such an Alice-and-Bob-style protocol description, as explained, for instance, in [18, 39, 44, 50].

The NSPK protocol can be described by the following sequence of message exchanges.

\[
\begin{align*}
\text{msg}_1 & : a \rightarrow b : (n_1) . \{ n_1; a \}^a_{K_i} \\
\text{msg}_2 & : b \rightarrow a : (n_2) . \{ n_1; n_2 \}^b_{K_a} \\
\text{msg}_3 & : a \rightarrow b : \{ n_2 \}^a_{K_b}
\end{align*}
\]

In this notation, \( a \) and \( b \) are variables identifying the principals playing in the different protocol roles (initiator and responder), \( n_1 \) and \( n_2 \) are variables representing the nonces created by these principals, and the arrows represent communication from the sender to the receiver. The parenthesized nonces prefixing the first two messages signify that these nonces are freshly generated before the message is sent. Moreover, it is assumed that the principals’ public keys have been distributed before the protocol starts (or else \( a \) and \( b \) would not be able to construct the messages). This can be straightforwardly expressed by appropriate instances of the axioms (N) and (PK).

Formalizing a protocol like the above involves defining the sequences of actions \((\text{send}, \text{rec}, \text{and fresh})\) taken by agents executing each protocol role. Specifically, given concrete principals \( A \) and \( B \) and fresh nonces \( N_1 \) and \( N_2 \), the role instantiations should correspond to the execution, by principal \( A \), of the sequence of actions \( \text{run}^{\text{init}}_A(A, B, N_1, N_2) \):

\[
(\text{fresh}(N_1) . \text{send}(\{ N_1; A \}^a_{K_B}, B) . \text{rec}(\{ N_1; N_2 \}^a_{K_A}) . \text{send}(\{ N_2 \}^a_{K_B}, B))
\]

and to the execution, by principal \( B \), of the sequence \( \text{run}^{\text{resp}}_B(A, B, N_1, N_2) \):

\[
(\text{rec}(\{ N_1; A \}^a_{K_B}) . \text{fresh}(N_2) . \text{send}(\{ N_1; N_2 \}^a_{K_A}, A) . \text{rec}(\{ N_2 \}^a_{K_B}))
\]

In general, an Alice-and-Bob-style protocol description \( P \) may involve \( j \) principal identifier variables \( a_1, \ldots, a_j \), corresponding to \( j \) distinct roles, and \( k \) fresh data variables \( f_1, \ldots, f_k \) (standing for the freshly created nonces, symmetric keys, or private asymmetric keys), and consist of a sequence \( (\text{msg}_1 \ldots \text{msg}_m) \) of message exchanges, each of the form

\[
(\text{msg}_q)~ a_s \rightarrow a_r : (f_{q_1}, \ldots, f_{q_t}) . M
\]

where \( M \) can include any of the principal identifiers and fresh data variables.

A protocol instantiation \( \sigma \) is a variable substitution such that each \( \sigma(a_i) \in \text{Princ} \) and each \( \sigma(f_i) \in \text{Nonces} \oplus \text{Sym} K \oplus \text{PrivK} \). Moreover, while the intruder can play different roles in a protocol instantiation, we require that honest agents do not play two roles in the same instantiation. Of course, this does not prevent the same honest agent from playing the same or other roles in other protocol instantiations. Hence, if \( \sigma(a_{i_1}), \sigma(a_{i_2}) \in \text{Hon} \) and \( \sigma(a_{i_1}) = \sigma(a_{i_2}) \) then \( i_1 = i_2 \).

We extend \( \sigma \) to messages, actions, sequences, formulas, and indices in the natural way. For example, \( \sigma(K_{a_i}) = K_{\sigma(a_i)} \). Each instantiation prescribes a concrete sequence of actions to be
executed by each participant in a protocol run: for each role $i$, we have the corresponding sequence $\text{run}^i = \text{msg}^i_1, \ldots, \text{msg}^i_m$ where

$$\text{msg}^i_q = \begin{cases} 
\langle \text{fresh}(f_{q_i}) \ldots \text{fresh}(f_{q_i}).\text{send}(M,a_r) \rangle & \text{if } i = s, \\
\langle \text{rec}(M) \rangle & \text{if } i = r, \\
\langle \rangle & \text{if } i \neq s \text{ and } i \neq r.
\end{cases}$$

If $\sigma(a_i) = A$, we write $\text{run}^i_A(\sigma) = \sigma(\text{run}^i)$. We can easily formalize the complete execution by principal $A$ of the run corresponding to role $i$ of the protocol, under the protocol instantiation $\sigma$. If $\text{run}^i_A(\sigma) = \langle \text{act}_1 \ldots \text{act}_n \rangle$ then we can formalize $A$’s execution by the local formula $\text{role}^i_A(\sigma)$:

$$\text{act}_n \land P(\text{act}_{n-1} \land \ldots \land P(\text{act}_1) \ldots).$$

In general, if we denote the set of all protocol instantiations by $\text{Inst}$, we can define the set $\text{Runs}^i_A$ of all possible concrete runs of principal $A$ in role $i$, and the set $\text{Runs}_A$ of all of $A$’s possible concrete runs in any of the $j$ roles:

$$\text{Runs}^i_A = \bigcup_{\sigma \in \text{Inst}} \{ \text{run}^i_A(\sigma) \mid \sigma(a_i) = A \in \text{Princ} \} \quad \text{and} \quad \text{Runs}_A = \bigcup_{i=1}^j \text{Runs}^i_A.$$  

It should be clear that $\mu, \xi \models [A][\text{role}^i_A(\sigma)]$ if and only if $A$ has just completed the required sequence of actions $\text{run}^i_A(\sigma)$ at $\xi$. Often, in examples, we will use $\pi = \langle a_1, \ldots, a_j \rangle$ and $\mathcal{I} = \langle f_1, \ldots, f_k \rangle$, and write $\text{run}^i_A(\sigma(\pi), \sigma(\mathcal{I}))$ instead of $\text{run}^i_A(\sigma)$, and $\text{role}^i_A(\sigma(\pi), \sigma(\mathcal{I}))$ instead of $\text{role}^i_A(\sigma)$.

In addition to the assumption that no honest agent ever plays two different roles in the same run, we also require that honest principals strictly follow the protocol. Therefore, if the local life-cycle of $A \in \text{Hon}$ is $e_1 \rightarrow_A e_2 \rightarrow_A e_3 \rightarrow_A \ldots$, we require that the corresponding (possibly infinite) sequence of actions

$$w(A) = \langle \alpha_A(e_1), \alpha_A(e_2), \alpha_A(e_3), \ldots \rangle$$

must be an interleaving of prefixes of sequences in $\text{Runs}_A$, but using distinct fresh data in each of them. Formally, we say that two distinct sequences of actions $w$ and $w'$ are independent provided that if $w_i = \text{fresh}(X)$ for some $i \leq |w|$ and some $X$, and $w'_j = \text{fresh}(Y)$ for some $j \leq |w'|$ and some $Y$, then $X \neq Y$. The requirement on protocol models can now be rigorously defined. For each $A \in \text{Hon}$, there must exist a set $W \subseteq \text{Runs}_A$ of pairwise independent sequences such that for every $i \leq |w(A)|$ it is possible to choose $w \in W$, $j \leq |w|$, and $t_1 < \cdots < i_j = i$ satisfying $w(A)_{t_k} = w_k$ for all $k$, where $k \leq j$. We will use the protocol name $P$ to denote the resulting set of models.

Note that this is similar to approaches such as [55], where the behavior of an honest agent $A$ is defined inductively so that the $i$th action of a sequence $w \in \text{Runs}_A$ can be executed only if the previous $i - 1$ actions have been executed. It is also similar to strand spaces [17, 62, 63] where essentially the same sequences of $\text{Runs}_A$ are used to model honest agents. In all cases, the intruder can act freely, according to the standard Dolev-Yao capabilities.

In the case of NSPK models, the life-cycle of each honest agent must be built by interleaving prefixes of sequences of the form $\text{run}^{\text{inst}}_A(A, B', N_1, N_2)$ or $\text{run}^{\text{resp}}_A(B', A, N_1, N_2)$, where no two such initiator runs can have the same $N_1$, no two responder runs can have the same $N_2$, and the $N_1$ of an initiator run must be different from the $N_2$ of any responder run.

### 3.4 Security goals

The aim of security protocol analysis is to prove (or disprove) the correctness of a protocol with respect to the security goals that the protocol should achieve. For instance, the secrecy of the critical data exchanged during a protocol’s execution is one such goal. In addition, an honest principal running the protocol may wish to authenticate the identities of its protocol partners based
on the messages he receives. There are many approaches to specifying secrecy and authentication in the literature, depending in part on the underlying model used. However, the various approaches usually agree on the general picture. Below, we show how to formulate secrecy and authentication goals for protocols in the general case and use the NSPK protocol as an illustration. As usual, given a protocol and a security goal, we call an attack any protocol model \( \mu \) and state \( \xi \) for which the formula expressing the goal does not hold. Let us start with secrecy.

### 3.4.1 Secrecy

We formalize that the messages in a finite set \( S \) will remain a shared secret between the participants \( A_1, \ldots, A_j \) after the complete execution of a protocol under the instantiation \( \sigma \), with each \( \sigma(a_i) \in \text{Princ} \), by the formula \( \text{secr}_S(\sigma) \):

\[
\Bigwedge_{i=1}^{j} \, @A_i[P \role^j_{A_i}(\sigma)] \Rightarrow \bigwedge_{B \in \text{Princ} \setminus \{A_1, \ldots, A_j\}} \bigwedge_{M \in S} \, @B[\neg \text{knows}(M)].
\]

Of course, this property can be expected to hold only in particular situations. Assume that all the participants are honest, that is, each \( A_i \in \text{Hon} \). One might then expect that the “critical” fresh data generated during the run will remain a secret shared only by the participating principals. Indeed, being honest, they will not reuse this fresh data in subsequent protocol runs. Using the logic, we can check the property \( \text{secr}_{\sigma(F)}(\sigma) \) for the relevant set of fresh data variables \( F \subseteq \{f_1, \ldots, f_k\} \). As before, we sometimes write \( \text{secr}_{\sigma(F)}(\sigma(\pi), \sigma(\mathcal{J})) \) instead of \( \text{secr}_{\sigma(F)}(\sigma) \).

In the case of the NSPK protocol, this amounts to requiring that \( \text{secr}_{\{N_1, N_2\}}(A, B, N_1, N_2) \) holds, with \( A \) and \( B \) both honest.

### 3.4.2 Authentication

There are many possible notions of authentication, e.g., the authentication hierarchy of [43]. In most cases, authentication formalizes some kind of correspondence property between the messages an agent receives in a protocol run and the messages that the other participants of the same run are supposed to have sent. The typical (weak) authentication goal states that if an honest principal \( A \) completes his part of a run of a protocol in role \( i \), with certain partners and data, then it must be the case that these partners have actually sent to \( A \) the messages that \( A \) received.

Let \( \sigma \) be a protocol instantiation such that \( \sigma(a_i) = A \in \text{Hon} \) and \( \sigma(a_j) = B \in \text{Princ} \). Then the property that \( A \) authenticates \( B \) in role \( j \) at message \( q \) of the protocol can be defined in our logic by the formula \( \text{auth}^{i,j,q}_{A,B}(\sigma) \), which is

\[
@A[\role^{i}_{A}(\sigma)] \Rightarrow @B[P \send(\sigma(M), A)],
\]

assuming that the protocol message \( \text{msg}_q \) requires that \( a_j \) sends the message \( M \) to \( a_i \). We would therefore require \( \text{auth}^{i,j,q}_{A,B}(\sigma) \) to hold whenever message \( q \) is considered essential for authentication. As before, we sometimes write \( \text{auth}^{i,j,q}_{A,B}(\sigma(\pi), \sigma(\mathcal{J})) \) instead of \( \text{auth}^{i,j,q}_{A,B}(\sigma) \).

In the case of the NSPK protocol we can specify, for an honest principal \( A \) acting as initiator, the authentication of the responder \( B \) at message 2 using \( \text{auth}^{\text{Init},\text{Resp},2}_{A,B}(A, B, N_1, N_2) \) as

\[
@A[\role^{\text{Init}}_{A}(A, B, N_1, N_2)] \Rightarrow @B[P \send([N_1; N_2]\,_{K_A}, A)].
\]

Analogously, for an honest principal \( B \) acting as responder, the authentication of the initiator \( A \) at message 3 using \( \text{auth}^{\text{Resp},\text{Init},3}_{B,A}(A, B, N_1, N_2) \) is

\[
@B[\role^{\text{Resp}}_{B}(A, B, N_1, N_2)] \Rightarrow @A[P \send([N_2]\,_{K_B}, B)].
\]

This last property fails due to the man-in-the-middle attack on NSPK [42], as we show below.
4 A meta-level secrecy result and object-level applications

In this section, we show how DTL can be used as an object logic for both protocol verification and falsification. In particular, we show that the responder fails to authenticate the initiator in the NSPK protocol, formalized in the previous section. Afterwards, we verify Lowe’s corrected version of this protocol. For this proof, however, we leverage DTL as a metalogic and first prove a general metatheorem about sufficient conditions for data to remain secret during communication. Our verification of the corrected protocol follow as a direct application of this general result.

4.1 Secret data

The result that we will prove is a good example of the kind of meta-level property that any suitable network model should enjoy. We use the properties of secure messages proved in Section 3 to reason about secrecy in a protocol-independent way and afterwards prove a proposition about secrecy properties in protocol models.

Let \( S \subseteq \text{Msg} \) be a set of secret messages, that is, messages that should not be disclosed. The following lemma states that, given a group of principals \( G \subseteq \text{Princ} \), if all the fresh data in \( S \) originates from principals in \( G \), then the secrets in \( S \) will remain unknown outside of \( G \) as long as the principals in \( G \) only send \( S \)-secure messages.

**Lemma 4.1 (Secret Data)** Let \( S \) be a rational set of messages, \( G \subseteq \text{Princ} \) be a group of principals, and \( \mu \) be a network model such that

\[
\mu \models \forall_{\mathcal{G} \subseteq \text{Princ}} @_{\mathcal{A}} [\star \Rightarrow (\text{knows}(X) \lor \text{fresh}(X)))] \quad \text{for each } X \in (\text{Nonces} \cup \text{SymK} \cup \text{PrivK}) \cap S.
\]

(i) Given a global state \( \xi \) of \( \mu \) with

\[
\mu, \xi \models \bigwedge_{B \in \text{Princ}\setminus G} @_{\mathcal{B}} [\neg \text{knows}(M)] \quad \text{for every } M \notin S\text{-Sec}
\]

(ii) then, for every global state \( \xi' \supseteq \xi \) of \( \mu \), either

\[
\mu, \xi' \models \bigwedge_{B \in \text{Princ}\setminus G} @_{\mathcal{B}} [\neg \text{knows}(M)] \quad \text{for every } M \notin S\text{-Sec}
\]

(iii) or else there exists \( M \notin S\text{-Sec} \) such that

\[
\mu, \xi' \models \bigvee_{\mathcal{A} \in \mathcal{G}, C \in \text{Princ}} @_{\mathcal{A}} [P_{e} \text{send}(M,C)].
\]

(iv)

Proof: We assume (i) and (ii), and prove either (iii) or (iv) for every global state \( \xi' \supseteq \xi \) of \( \mu \). The proof proceeds by induction, using the global invariance rule of Proposition 2.3. The rule’s base case (1), with \( \xi' = \xi \), is trivial, as in this case (iii) coincides (ii). We turn to the step case (2), and must show that assuming, by induction hypothesis, that either (iii) or (iv) hold for \( \xi' \), then (iii) or (iv) must also hold for any extended global \( \xi' \cup \{e\} \).

By definition, if (iv) holds for \( \xi' \), then it also holds for \( \xi' \cup \{e\} \). Thus, we are left with proving that if (iii) holds for \( \xi' \), then (iii) or (iv) hold for \( \xi' \cup \{e\} \). Suppose then that (iii) holds for \( \xi' \) but not for \( \xi' \cup \{e\} \), that is \( \mu, \xi' \cup \{e\} \models @_{\mathcal{B}} [\text{knows}(M)] \) for some \( S \)-insecure message \( M \) and some \( B \notin G \). Then, by Lemma 2.1, it must be the case that \( e \in Ev_{B} \), and so the local states of all other principals do not change (see axiom (P5)). Moreover, \( \alpha_{B}(e) \) cannot be a sending action since this would not change the knowledge of principal \( B \) (see condition (K)). If \( \alpha_{B}(e) \) was either rec(\( M' \)) or spy(\( M' \)) then, using Proposition 3.9 and the assumption that (iii) holds for \( \xi' \), it would follow that \( M' \notin S\text{-Sec} \). However, since \( M' \) must have been previously sent to the channel (by the axioms (P3), (C2), (P4), and (C1)), axiom (P1) implies that such a message could only have been sent by some \( A \in G \). Hence, one would have \( \mu, \xi' \cup \{e\} \models \bigvee_{\mathcal{A} \in \mathcal{G}, C \in \text{Princ}} @_{\mathcal{A}} [P_{e} \text{send}(M',C)] \), i.e., (iv) would hold for \( \xi' \cup \{e\} \). The only remaining possibility is a fresh action. However, it cannot be \( \text{fresh}(X) \) for some \( X \in S \), independently of whether \( X \) is a nonce, a symmetric key, or a private key, as this, together with the freshness conditions ((F1-2), (K7), (aKey1), (sKey1.1 – 2)) and the fact that \( B \notin G \), would contradict condition (i), and the result follows. □
Note that the set $\text{Msg} \setminus S\text{-Sec}$ of $S$-insecure messages corresponds to what is called an ideal in the context of strand spaces [62]. However, here we have defined it in a more general setting that also includes composed keys and hashing. Similarly, the set $S\text{-Sec}$ of $S$-secure messages corresponds to a coideal in the terminology of [27, 47].

Lemma 4.1 above is a general, protocol-independent result about the network data flow. It can be used to reason about secrecy properties in protocol models and is similar to results obtained for PCL [56], which in turn generalize those found in [16, 27, 47], e.g., to include composed keys and hashing. Indeed, under reasonable conditions, the secrecy of freshly generated data can easily be seen to hold. Recall that we assume that each principal $A$ initially has a private key, denoted by $K^{-1}_A$, and that each pair of principals $A$ and $B$ initially shares a symmetric key, denoted by $K_{AB}$, subject to the key axioms (aKey1), (aKey2), (sKey1.1), (sKey1.2) and (sKey2) given in Section 3.3. If the protocol at hand does not require the existence of some of these initially distributed keys, then the result still holds if we simply remove the unused keys from the set $S$.

**Proposition 4.2 (Secrecy)** A protocol guarantees secrecy $\text{secr}_{\sigma(F)}(\sigma)$ for an instantiation $\sigma$ of the fresh data $\sigma(F)$ generated in a protocol run by honest participants $\sigma(a_1) = A_1$, ..., $\sigma(a_j) = A_j$ provided that all the messages ever sent by $A_1, A_2, ..., A_j$ in any protocol run are $S$-Secure, for $S = (\{K^{-1}_A | 1 \leq i \leq j\} \cup \{K_{A_i A_k} | 1 \leq i, k \leq j, i \neq k\} \cup \sigma(F))$.

**Proof:** The result follows from Lemma 4.1 using $G = \{A_1, ..., A_j\}$. To begin with, $S$ is a rational set. Let $\mu$ be a network model and $\xi$ a global state and assume that $\mu, \xi \models \bigwedge_{i=1}^{j} \Delta_{A_i}^0 [P_2 \ \text{role}^i_{A_i}(\sigma)]$. Condition (i) of the lemma follows as all the corresponding roles of the protocol have been completed and therefore all fresh data in $\sigma(F)$ is generated in $\mu$ among the principals in $G$. Moreover, for the initial state $\emptyset$, clearly, no principal outside $G$ knows $S$-insecure messages. For fresh nonces, condition (ii) follows directly from axioms (F1 – 2). For private and shared keys, condition (ii) follows from the axioms (aKey2) and (sKey2). Using the lemma, we then have that (iii) or (iv) must hold for any global state of $\mu$. However, the assumption that $A_1, A_2, ..., A_j$ only send $S$-secure messages rules out the possibility that condition (iv) of Lemma 4.1 ever holds. Hence, (iii) must be the case also at $\xi$, i.e., $\mu, \xi \models \bigwedge_{B \in \text{Princ} \setminus G} [P_B \ \text{knows}(M)]$ for every $M \notin S\text{-Sec}$. It follows that $\mu, \xi \models \bigwedge_{B \in \text{Princ} \setminus \{A_1, ..., A_j\}} \bigwedge_{M \in \sigma(F)} [P_B \ \text{knows}(M)]$, and the secrecy property holds.

Note that our assumption that all the messages sent by $A_1, A_2, ..., A_j$ in any protocol run are $S$-secure corresponds to the notion of discreteness of [27, 47].

### 4.2 Object-level analysis of NSPK

We will now illustrate the use of DTL as an object logic using two well-known examples: the NSPK protocol and its corrected version by Lowe.

#### 4.2.1 Protocol falsification

Recall that in the previous section, we formalized the NSPK protocol and the property that the responder authenticates the initiator: $\text{auth}_{A,B}$, $\text{auth}_{A,B}$. We now show that this cannot be proved by presenting a model that falsifies it. The model formalizes the well-known man-in-the-middle-attack discovered by Lowe [42], where the intruder $Z$ makes $B$ believe he has run the protocol with $A$. It illustrates how concrete execution scenarios can be formalized as DTL models.

Figure 5 presents this model. It is straightforward to see that it satisfies all of the axioms of the channel model presented in Section 3. For example, (C1) and (P2) are globally satisfied because every in action is synchronized with a corresponding send action by the same principal and vice versa. Another example is (P1), which says that only known messages are sent. This holds for each of the 5 send actions. For example, for send2, the intruder must know $[\{N_1; A\}]_{K_Z}^{\sigma}$, which is the case because he previously received $[\{N_1; A\}]_{K_Z}^{\sigma}$, which he can decrypt, and because he also knows public keys, in particular $K_B$. Additionally, the antecedent of our authentication

18
We now show that the authentication property $\text{auth}_{BA}^{\text{Resp,Init},3}(A, B, N_1, N_2)$ for honest $B$, which failed for NSPK, actually holds for the NSL protocol \cite{42} given below:

\[
\begin{align*}
(m_{g_1}) & \quad a \rightarrow b : \langle m_1 \rangle_{K_a}^a \\
(m_{g_2}) & \quad b \rightarrow a : \langle m_2 \rangle_{K_b}^b \\
(m_{g_3}) & \quad a \rightarrow b : \langle m_2 \rangle_{K_a}^a
\end{align*}
\]

Again, we have two roles: an initiator role $\text{Init}$, represented by $a$, and a responder role $\text{Resp}$, represented by $b$. Given principals $A$ and $B$ and nonces $N_1$ and $N_2$, the role instantiations should correspond to the execution by principal $A$ of the sequence of actions $\text{run}_{A}^{\text{Init}}(A, B, N_1, N_2)$:

\[
\langle \text{fresh}(N_1).\text{send}([N_1; A]_{K_B}^a, B).\text{rec}([N_1; N_2; B]_{K_A}^a).\text{send}([N_2]_{K_B}^a) \rangle.
\]

and to the execution by principal $B$ of the sequence $\text{run}_{B}^{\text{Resp}}(A, B, N_1, N_2)$:

\[
\langle \text{rec}([N_1; A]_{K_B}^a).\text{fresh}(N_2).\text{send}([N_1; N_2; B]_{K_A}^a, A).\text{rec}([N_2]_{K_B}^a) \rangle.
\]

Figure 5: Man-in-the-middle attack to NSPK
Prior to showing in Proposition 4.5 that this protocol authenticates the initiator, we prove two lemmas. The first lemma allows us to conclude that if A is also honest and does not send the required message \( \{N_2\}_{K_a} \) to B, then no other agent would ever have the means to do it either.

**Lemma 4.3** Let \( A, B \in \text{Hon} \), \( \mu \) be a NSL model, and \( \xi \) a global state such that

\[
\mu, \xi \models \text{@B}[\text{fresh}(N_2) \land \text{F send}(\{N_1; N_2; B\}_{K_a}^\mu, A)].
\]

For every global state \( \xi' \supseteq \xi \), if

\[
\mu, \xi' \not\models \text{@A}[P_a \text{send}(\{N_2\}_{K_a}^\mu, B)]
\]

then, for every \( C, D, E \in \text{Princ} \) with \( E \not\in \{A, B\} \), and every message \( M \) which is not \( \{N_2, K_A^{-1}\}\)-secure, or \( M \) contains \( N_2 \) outside a submessage \( \{N_1; N_2; B\}_{K_a}^\mu \), the following holds:

\[
\mu, \xi' \not\models \text{@CB}[P_a \text{in}(C, M, D)] \text{ and } \mu, \xi' \not\models \text{E}[\text{knows}(M)].
\]

**Proof:** We prove this with the help of Lemma 4.1. Let \( G = \{A, B\} \) and \( S = \{N_2, K_A^{-1}\} \cup \{M \mid M \text{ contains } N_2 \text{ outside a submessage } \{N_1; N_2; B\}_{K_a}^\mu\} \). It is not difficult to check that \( S \) is a rational set. Indeed, \( \text{parts}(N_2) = \emptyset \) and, for each of the remaining messages \( M \), either \( N_2 \in \text{parts}(M) \) and \( \text{parts}(M) \cap S \neq \emptyset \) or \( M' \in \text{parts}(M) \) for some message \( M' \) that contains \( N_2 \) outside a submessage \( \{N_1; N_2; B\}_{K_a}^\mu \). But, in this case, \( M' \in S \) and \( \text{parts}(M) \cap S \neq \emptyset \).

Next, we establish condition (i) of Lemma 4.1. The only relevant atoms in \( S \) are \( N_2 \) and \( K_A^{-1} \). As \( \mu, \xi \models \text{@B}[\text{fresh}(N_2)] \), it follows that \( \mu \models \text{@B}[\ast \Rightarrow \text{F fresh}(N_2)] \). For \( K_A^{-1} \), it follows from axiom (aKey1) that \( \mu \models \text{@A}[\ast \Rightarrow \text{knows}(K_A^{-1})] \).

Now, let \( E \not\in G \) and suppose that \( \mu, \xi \models \text{E}[\text{knows}(M)] \). From \( \mu, \xi \models \text{B}[\text{fresh}(N_2)] \) and the freshness axioms (F1-2) it follows not only that \( M \) is \( \{N_2\}_S \)-secure, but also that \( M \) does not even contain \( N_2 \). Furthermore, axiom (aKey2) and the honesty of \( A \) guarantee that all the messages ever in the channel, or known by some principal other than \( A \), are \( \{K_A^{-1}\}_S \)-secure. Thus, the message \( M \) is \( S \)-secure. This implies that condition (ii) of Lemma 4.1 holds, which tells us that for every global state \( \xi' \supseteq \xi \) of \( \mu \) either condition (iii) or condition (iv) of Lemma 4.1 must hold.

We proceed to show that condition (iv) cannot hold. Concretely, we prove by global induction on \( \xi' \supseteq \xi \) that if \( \mu, \xi' \not\models \text{@A}[P_a \text{send}(\{N_2\}_{K_a}^\mu, B)] \) then condition (iii) of Lemma 4.1 holds for \( \xi' \) and condition (iv) of Lemma 4.1 does not. The base case with \( \xi' = \xi \) is simple: the same argument that we used to show that condition (ii) of Lemma 4.1 holds, together with axiom (P1), establishes that condition (iv) cannot hold at \( \xi \). For the induction step, let us assume that \( \xi' \cup \{e\} \supseteq \xi' \supseteq \xi \) are global states and that, by the induction hypothesis, if \( \mu, \xi' \not\models \text{@A}[P_a \text{send}(\{N_2\}_{K_a}^\mu, B)] \) then (iii) holds for \( \xi' \) and (iv) does not. Since, by Lemma 4.1, we know that (iii) or (iv) must hold, it suffices to show that if \( \mu, \xi' \cup \{e\} \not\models \text{@A}[P_a \text{send}(\{N_2\}_{K_a}^\mu, B)] \) then (iv) cannot hold for \( \xi' \cup \{e\} \).

We must then consider, in turn, the two relevant cases for condition (iv) to hold, that is, \( e \in Ev_A \) and \( e \in Ev_B \).

**Case (1):** If \( e \in Ev_A \), we analyze the possible sending actions \( \alpha_A(e) \).

1. If \( \alpha_A(e) = \text{send}(\{N_1^*; A\}_{K_X}^\mu, X) \), it must be in a prefix

\[
(\text{fresh}(N_1^*).\text{send}(\{N_1^*; A\}_{K_X}^\mu, X))
\]

of an initiator run of the protocol with some \( X \in \text{Princ} \). As \( A \) freshly generated \( N_1^* \), and \( B \) freshly generated \( N_2 \), the freshness axioms (F1-2) guarantee that \( N_1^* \neq N_2 \). Therefore, the message \( M = \{N_1^*; A\}_{K_X}^\mu \in S\text{-Sec} \).

2. If \( \alpha_A(e) = \text{send}(\{N_1^*; N_2^*; A\}_{K_X}^\mu, X) \), it must be in a prefix

\[
(\text{rec}(\{N_1^*; X\}_{K_X}^\mu).\text{fresh}(N_2^*).\text{send}(\{N_1^*; N_2^*; A\}_{K_X}^\mu, X))
\]

20
of a responder run of the protocol with some \( X \in \text{Princ} \). As \( A \) freshly generated \( N_2^* \), and \( B \) freshly generated \( N_2 \), the freshness axioms (F1-2) guarantee that \( N_2^* \neq N_2 \). Moreover, \( A \) first received \([N_1^*; X]^a_{K_A} \) from the channel, and axioms (P3), (C2), (C1), (P1) together with the induction hypothesis guarantee that \( N_1^* \neq N_2 \). Therefore, \( M = [N_1^*; N_2^*; A]^a_{K_X} \in S-\text{Sec} \) as in case (1.1) above.

(1.3) If \( \alpha_A(e) = \text{send}([N_2^*]^a_{K_X}, X) \), it must be in a complete initiator run

\[
(fresh(N_1^*).send([N_1^*; A]^a_{K_X}, X).rec([N_1^*; N_2^*; X]^a_{K_A}).send([N_2^*]^a_{K_X}, X))
\]

of the protocol with some \( X \in \text{Princ} \). If, by absurdity, \( N_2^* = N_2 \), then as \( A \) would first receive \([N_1^*; N_2; X]^a_{K_A} \) from the channel, it would follow from axioms (P3), (C2), (C1), (P1) and the induction hypothesis that \( N_1^* = N_1 \) and \( X = B \). But this would contradict \( \mu, \xi \cup \{ e \} \upharpoonright @_A \text{send}([N_2^*]^a_{K_B}, B) \). Hence, \( N_2^* \neq N_2 \) and \( M = [N_2^*]^a_{K_X} \in S-\text{Sec} \).

Case (2): If \( e \in Ev_B \), we also analyze the possible sending actions \( \alpha_B(e) \).

(2.1) If \( \alpha_B(e) = \text{send}([N_1^*; B]^a_{K_X}, X) \), it must be in a prefix

\[
(fresh(N_1^*).send([N_1^*; B]^a_{K_X}, X))
\]

of an initiator run of the protocol with some \( X \in \text{Princ} \). As \( B \) freshly generated both \( N_1^* \) and \( N_2 \), but \( N_2 \) in a responder run, \( B \)'s honesty guarantees that \( N_1^* \neq N_2 \). Hence, \( M = [N_1^*; B]^a_{K_X} \in S-\text{Sec} \).

(2.2) If \( \alpha_B(e) = \text{send}([N_1^*; N_2^*; B]^a_{K_X}, X) \), it must be in a prefix

\[
(rec([N_1^*; X]^a_{K_B}).fresh(N_2^*).send([N_1^*; N_2^*; B]^a_{K_X}, X))
\]

of a responder run of the protocol with some \( X \in \text{Princ} \). If \( N_2^* = N_2 \), since \( \mu, \xi \upharpoonright @_B \text{fresh}(N_2) \land F \text{send}(\{N_1; N_2; B]^a_{K_X}, A) \), it follows from \( B \)'s honesty that also \( N_1^* = N_1 \) and \( X = A \). Clearly, \( M = [N_1; N_2; B]^a_{K_X} \in S-\text{Sec} \). In contrast, if \( N_2^* \neq N_2 \) then, as \( B \) first received \([N_1^*; X]^a_{K_B} \), the axioms (P3), (C2), (C1), (P1) together with the induction hypothesis guarantee that \( N_1^* \neq N_2 \). Thus, again, \( M = [N_1^*; N_2^*; A]^a_{K_X} \in S-\text{Sec} \).

(2.3) If \( \alpha_B(e) = \text{send}([N_2^*]^a_{K_X}, X) \), it must be in a complete initiator run

\[
(fresh(N_1^*).send([N_1^*; B]^a_{K_X}, X).rec([N_1^*; N_2^*; X]^a_{K_B}).send([N_2^*]^a_{K_X}, X))
\]

of the protocol with some \( X \in \text{Princ} \). Note that, \( N_2^* = N_2 \) is impossible, as \( B \) would first receive \([N_1^*; N_2; X]^a_{K_B} \) from the channel, and axioms (P3), (C2), (C1), (P1) together with the induction hypothesis would yield a contradiction. Therefore, \( N_2^* \neq N_2 \), and \( M = [N_2^*]^a_{K_X} \in S-\text{Sec} \).

Thus, (iv) never holds, (iii) always holds, and the statement follows.

Note that an analog of this lemma fails for the original NSPK protocol. The proof fails at case (1.3) because the message \([N_1; N_2]^a_{K_A} \) can be understood by \( A \) as belonging to a run executed with an agent different from \( B \), namely the intruder, as we saw in the previous countermodel.

Our second lemma allows us to conclude that if \( A = Z \) is the initiator, then it will never be able to trick an honest principal into sending the message \([N_2]^a_{K_B} \) to \( B \) in another run of the protocol. Namely, the lemma shows that if \( B \) is playing the responder in another protocol run initiated by an honest agent, then it will not mix the relevant data of the two runs.

**Lemma 4.4** Let \( B \in \text{Hon} \), \( \mu \) be a NSL model, and \( \xi \) a global state such that

\[
\mu, \xi \upharpoonright @_B \text{fresh}(N_2) \land F \text{send}(\{N_1; N_2; B]^a_{K_Z}, Z),
\]
and let also $C \in \text{Hom}$, and $\xi'$ a global state such that

$$\mu, \xi' \vDash @_C[\text{fresh}(N_1') \land F \text{ send}(\{N_1' ; C\}^{a}_{KB}, B)].$$

For every global state $\xi'' \supseteq \xi'$, every $D, E, F \in \text{Princ}$ with $F \notin \{C, B\}$, and every message $M'$ which is not $\{N_1', K_B^{-1}, K_C^{-1}\}$-secure, or $M'$ contains $\{N_1'; N_2; B\}^{a}_{KB}$, or $\{N_1'; X\}^{a}_{KV}$ with $X \neq C$ or $Y \neq B$, or $\{N; N_1'; X\}^{a}_{KB}$, with any $N, X, Y$, the following holds:

$$\mu, \xi'' \not\vDash @_C[\text{in}(D, M', E)] \text{ and } \mu, \xi'' \not\vDash @_F[\text{knows}(M')].$$

Proof: We reuse Lemma 4.1, now with $C = \{B, C\}$ and $S = \{N_1'; K_B^{-1}, K_C^{-1}\} \cup \{M' | M' \text{ contains } \{N_1'; N_2; B\}^{a}_{KB}, \text{ or } \{N_1'; X\}^{a}_{KV}, \text{ with } X \neq C \text{ or } Y \neq B, \text{ or } \{N; N_1'; X\}^{a}_{KB}, \text{ for any } N, X, Y\}. The proof that $S$ is a rational set and that conditions (i) and (ii) of Lemma 4.1 hold for $\xi''$ is similar to that of Lemma 4.3. Therefore, Lemma 4.1 tells us that for every global state $\xi'' \supseteq \xi'$ of $\mu$ either condition (iii) or condition (iv) of Lemma 4.1 must hold.

We proceed to show that condition (iv) cannot hold. Concretely, we prove by global induction on $\xi'' \supseteq \xi'$ that condition (ii) of Lemma 4.1 holds for $\xi''$ and condition (iv) of Lemma 4.1 does not hold. The base case with $\xi'' = \xi'$ is immediate. For the induction step, let us assume that $\xi'' \cup \{e\} \supseteq \xi'' \supseteq \xi'$ are global states and that, by induction hypothesis, (iii) holds for $\xi''$ and (iv) does not. Since, by Lemma 4.1, we know that (iii) or (iv) must hold, it suffices to show that (iv) cannot hold for $\xi'' \cup \{e\}$. Consequently, in turn, the two relevant cases for condition (iv) to hold, that is, $e \in Ev_B$ and $e \in Ev_C$.

Case (1): If $e \in Ev_B$, we analyze the possible sending actions $\alpha_B(e)$.

(1.1) If $\alpha_B(e) = \text{send}(\{N_1'; B\}^{a}_{KB}, X)$, it must be in a prefix

$$(\text{fresh}(N_1'), \text{send}(\{N_1'; B\}^{a}_{KB}, X))$$

of an initiator run of the protocol with some $X \in \text{Princ}$. As $B$ freshly generated $N_1'$, and $C$ freshly generated $N_1'$, the freshness axioms (F1-2) guarantee that $N_1' \neq N_1$. Thus, the message $M' = \{N_1'; B\}^{a}_{KB}$ is in $S$-Sec.

(1.2) If $\alpha_B(e) = \text{send}(\{N_1'; N_2; B\}^{a}_{KB}, X)$, it must be in a prefix

$$(\text{rec}(\{N_1'; X\}^{a}_{KB}), \text{fresh}(N_1'), \text{send}(\{N_1'; N_2; B\}^{a}_{KB}, X))$$

of a responder run of the protocol with some $X \in \text{Princ}$. As $B$ freshly generated $N_2'$, and $C$ freshly generated $N_1'$, the freshness axioms (F1-2) guarantee that $N_2' \neq N_1'$. Moreover, as $B$ first receives $\{N_1'; X\}^{a}_{KB}$, axioms (P3), (C2), (C1), (P1) together with the induction hypothesis guarantee that either $N_1' \neq N_1'$, or $N_1' = N_1$ and $X = C$. In the former case, it is immediate that $M' = \{N_1'; N_2; B\}^{a}_{KB}$ is S-secure. In the latter case, $N_2' = N_2$ is impossible, because $\mu, \xi \vDash @_B[\text{fresh}(N_2') \land F \text{ send}(\{N_1'; N_2; B\}^{a}_{KB}, Z)]$ and $B$’s honesty would necessitate $N_2' = N_1$ and $X = Z$, which contradicts $C$’s honesty. Therefore, $N_2' \neq N_2$ and the message $M' = \{N_1'; N_2; B\}^{a}_{KB}$ is S-secure.

(1.3) If $\alpha_B(e) = \text{send}(\{N_2'; B\}^{a}_{KB}, X)$, it must be in a complete initiator run

$$\langle \text{fresh}(N_2'), \text{send}(\{N_1'; B\}^{a}_{KB}, X), \text{rec}(\{N_1'; N_2; X\}^{a}_{KB}), \text{send}(\{N_2'; B\}^{a}_{KB}, X) \rangle$$

of the protocol with some $X \in \text{Princ}$. As $B$ will first receive $\{N_1'; N_2; X\}^{a}_{KB}$, axioms (P3), (C2), (C1) and (P1) along with the induction hypothesis guarantee that $N_2' \neq N_1'$. Thus, the message $M' = \{N_2'; B\}^{a}_{KB}$ is S-secure.

Case (2): If $e \in Ev_C$, we also analyze the relevant sending actions $\alpha_C(e)$.

(2.1) If $\alpha_C(e) = \text{send}(\{N_1'; C\}^{a}_{KB}, X)$, it must be in a prefix

$$(\text{fresh}(N_1'), \text{send}(\{N_1'; C\}^{a}_{KB}, X)).$$

If $N_0' \neq N_1'$ then $M' = \{N_0'; C\}^{a}_{KB}$ is S-secure. In contrast, if $N_0' = N_1'$, as $\mu, \xi' \vDash @_C[\text{fresh}(N_1') \land F \text{ send}(\{N_1'; C\}^{a}_{KB}, B)]$ it follows from $C$’s honesty that also $X = B$, and the message $M' = \{N_1'; C\}^{a}_{KB}$ is S-secure.
\(2.2\) If \(\alpha_c(e) = \text{send}(\langle N_1^2; N_2^2; C \rangle^{a}_{K_X}, X)\), it must be in a prefix
\[
\langle \text{rec}(\langle N_1^2; X \rangle^{a}_{K_C}).\text{fresh}(N_2^2).\text{send}(\langle N_1^2; N_2^2; C \rangle^{a}_{K_X}, X) \rangle
\]
of a responder run of the protocol with some \(X \in \text{Princ}\). As \(C\) freshly generates \(N_1^2\) and \(N_2^2\), but \(N_1^2\) in an initiator run, \(C\)'s honesty ensures that \(N_2^2 \neq N_1^2\). Moreover, since \(C\) first received \(\langle N_1^2; X \rangle^{a}_{K_C}\) from the channel, the axioms \((\text{P3})\), \((\text{C2})\), \((\text{C1})\) and \((\text{P1})\) along with the induction hypothesis guarantee that \(N_1^2 \neq N_1^2\). Therefore, the message \(M' = \langle N_1^2; N_2^2; C \rangle^{a}_{K_X}\) is S-secure.

\(2.3\) If \(\alpha_c(e) = \text{send}(\langle N_2^2 \rangle^{a}_{K_X}, X)\), it must be in a complete initiator run
\[
\langle \text{fresh}(N_1^2).\text{send}(\langle N_1^2; C \rangle^{a}_{K_X}, X).\text{rec}(\langle N_2^2; N_1^2 \rangle^{a}_{K_C}.\text{send}(\langle N_2^2 \rangle^{a}_{K_X}, X)) \rangle
\]
of the protocol with some \(X \in \text{Princ}\). The proof is similar to step (1.3) above.

Thus, (iv) never holds, (iii) always holds, and the statement follows. \(\square\)

We now proceed to the main result: the responder authenticates the initiator. Recall from Section 3.3 that given an honest principal \(B\), \(\text{auth}_{B,A}^{\text{Resp,Init,3}}(A, B, N_1, N_2)\) corresponds to
\[
@_B[\text{role}_B^{\text{Resp}}(A, B, N_1, N_2)] \Rightarrow @_A[\text{P}_0 \text{send}(\langle N_2 \rangle^{a}_{K_B}, B)],
\]
where \(\text{role}_B^{\text{Resp}}(A, B, N_1, N_2)\) is
\[
\text{rec}(\langle N_2 \rangle^{a}_{K_B}) \land \text{P}(\text{send}(\langle N_1; N_2; B \rangle^{a}_{K_A}, A) \land \text{P}(\text{fresh}(N_2) \land \text{P}(\text{rec}(\langle N_1; A \rangle^{a}_{K_B})))).
\]

**Proposition 4.5** \(\text{NSL} \vdash \text{auth}_{B,A}^{\text{Resp,Init,3}}(A, B, N_1, N_2)\) for \(A \in \text{Princ}, B \in \text{Hon},\) and \(N_1\) and \(N_2\) arbitrary distinct nonces.

**Proof:** Let \(\mu\) be an NSL model, \(\xi\) a global state such that \(\mu, \xi \models @_B[\text{role}_B^{\text{Resp}}(A, B, N_1, N_2)]\). Recall that \(B\) is honest. We have two cases: either (1) \(A\) is also honest or (2) \(A\) is actually \(Z\).

**Case (1):** Assume, by absurdity, that \(\mu, \xi \not\models @_A[\text{P}_0 \text{send}(\langle N_2 \rangle^{a}_{K_B}, B)]\). Since both \(A\) and \(B\) are honest, we can consider \(\xi'\), where \(\xi' \leq \xi\) and \(\mu, \xi' \not\models @_B[\text{fresh}(N_2)]\), and use Lemma 4.3 to conclude that \(\mu, \xi \not\models @_A[\text{P}_0 \text{send}(\langle N_2 \rangle^{a}_{K_B}, B)]\). However, given axioms \((\text{P3})\) and \((\text{C2})\), this contradicts the assumption that \(\mu, \xi \models @_A[\text{role}_B^{\text{Resp}}(A, B, N_1, N_2)]\), as in particular it must be the case that \(\mu, \xi \not\models @_A[\text{rec}(\langle N_2 \rangle^{a}_{K_B})]\). Hence, \(\mu, \xi \models @_A[\text{P}_0 \text{send}(\langle N_2 \rangle^{a}_{K_B}, B)]\).

**Case (2):** If \(A = Z\) then \(B\)'s run is actually
\[
\langle \text{rec}(\langle N_1; Z \rangle^{a}_{K_Z}), \text{fresh}(N_2).\text{send}(\langle N_1; N_2; B \rangle^{a}_{K_Z}, Z).\text{rec}(\langle N_2 \rangle^{a}_{K_B}) \rangle.
\]

We must show that \(\mu, \xi \models @_Z[\text{P}_0 \text{send}(\langle N_2 \rangle^{a}_{K_B}, B)]\). From axioms \((\text{P3})\) and \((\text{C1-2})\), we know that \(\mu, \xi \models @_C[\text{P}_0 \text{send}(\langle N_2 \rangle^{a}_{K_B}, B)]\). Hence, it suffices to prove that \(Z\) cannot trick an honest principal \(C\) into sending the message \(\langle N_2 \rangle^{a}_{K_B}\) to \(B\). We can exclude the case when \(C = B\), since in protocol models no honest principal sends messages to himself. Now, an honest \(C \neq B\) will only send such a message in an initiator role, of the form
\[
\langle \text{fresh}(N_1^C).\text{send}(\langle N_1^C; C \rangle^{a}_{K_C}, \text{rec}(\langle N_1^C; N_2; B \rangle^{a}_{K_C})) \rangle.
\]

But clearly only \(B\) himself, or \(Z\), would send \(\langle N_1^C; N_2; B \rangle^{a}_{K_C}\). Moreover, \(B\) can be excluded because his honesty guarantees that he will not use \(N_2\) but rather a fresh value, according to the definition of protocol models or else it must be the case that \(N_1^C = N_1\) and \(C = Z\), which contradicts the honesty of \(C\). Moreover, if for any \(\xi' \subseteq \xi\) we have indeed that \(\mu, \xi' \models @_C[\text{fresh}(N_1^C) \land \text{F send}(\langle N_1^C; C \rangle^{a}_{K_C}, B)]\), then Lemma 4.4 guarantees that \(\mu, \xi' \models @_Z[\text{knows}(\langle N_1^C; N_2; B \rangle^{a}_{K_B})]\), and therefore \(C\) would never receive \(\langle N_1^C; N_2; B \rangle^{a}_{K_B}\) and thus could not complete the run. We can therefore conclude that \(\mu, \xi \models @_Z[\text{P}_0 \text{send}(\langle N_2 \rangle^{a}_{K_B}, B)]\). \(\square\)
As NSPK and NSL are well studied, our results are not surprising and the proof ideas are similar to those found, e.g., in [55]. However, our development illustrates well how we can use our channel model to give semantic proofs and how we can exploit metatheoretic properties, all within DTL. Other security protocols can be falsified or verified similarly, using DTL as an object logic. We now turn to the main strength of DTL: its use as a metalogic to relate different models for security protocols and to prove further metatheorems about the models themselves.

5 Meta-level model analysis

In this section, we present two concrete examples of metareasoning. First, we prove the equivalence of two models for guaranteeing message-origin authentication. Second, we relate channel-based and intruder-centric models, showing that it is sufficient to consider models with a single intruder who controls the network.

5.1 Message-origin authentication

The following example is of a different nature than the last one. We now show how DTL can be used to relate models at different levels of abstraction. Establishing such formal relationships is a central paradigm in Information Security and is used, for example, when carrying out simulation proofs to show that concrete cryptographic operations implement a given ideal functionality. Our example is centered around message-origin authentication: ensuring that a message purported to come from an agent really originated with the agent. We will use DTL to study the relationship between two models designed to guarantee message-origin authentication.

1. An abstract model TTP where principals may use a special “logged” channel T controlled by a trusted third party. This channel logs all incoming messages and issues evidence of their origin to the recipients.

2. A concrete model DS that is closer to a possible realization of an authentic channel. Communication takes place in DS over a public channel, but principals digitally sign the messages they send so that their signatures can be verified by the recipients.

We investigate several relationships between these two models by exploring transformations of their corresponding DTL models, along with translations of their properties. By abstracting away details of the communication media, we prove that the two models are equivalent under mild assumptions about the nature of message-origin authentication.

5.1.1 TTP: trusted third party logging

In this model, we extend the channel-based model CB of Section 3 with an additional communication medium T, representing the logged channel and controlled by a trusted third party. Principals can choose to send or receive messages either through the public channel or T. Messages exchanged through T are logged by the trusted third party, who issues evidence of their origin to the recipients. Hence, all principals are augmented with actions for communicating using T and with state propositions that provide evidence of origin for the messages received from T.

Recall that we consider fixed a network signature \( \langle \text{Princ}, \text{Num} \rangle \). The signature \( \Sigma_{\text{TTP}} = (\text{Princ} \uplus \{ \text{Ch}, T \}, \text{Act}, \text{Prop}) \) is such that for each \( A \in \text{Princ} \) we have that

- \( \text{Act}_A \) is composed of
  - send\((M, B)\), rec\((M)\), spy\((M)\), and fresh\((X)\), as in \( \Sigma_{\text{CB}} \),
  - send\(_T(M, B)\): sending message \( M \) to \( B \) via \( T \),
  - rec\(_T(B, M)\): receiving from \( T \) message \( M \) originating from \( B \),
  - spy\(_T(B, M)\): eavesdropping in \( T \) message \( M \) originating from \( B \).
• Prop$_A$ includes the state propositions
  - $knows(M)$, as in $\Sigma_{CB}$,
  - $evid(B, M)$: evidence was obtained from $T$ that message $M$ originates from $B$;

and for $Ch$, as in $\Sigma_{CB}$, and also for $T$, we have

• Act$_{Ch} = Act_T$, where both consist of the actions $in(A, M, B)$, $out(A, M, B)$, and $leak$;
• Prop$_{Ch} = Prop_T = \emptyset$.

We use $L_{TTP}$ to denote the DTL language over the signature $\Sigma_{TTP}$.

The axiomatization of $TTP$ includes the axioms $(F1–F2)$, $(C1–C4)$, and $(P1–P6)$ of $CB$, from Section 3. It also includes corresponding versions of the channel axioms for $T$, namely

\begin{align*}
(T1) & \quad @_T[in(A, M, B) \Rightarrow_A send_T(M, B)] \\
(T2) & \quad @_T[out(A, M, B) \Rightarrow P \ in(A, M, B)] \\
(T3) & \quad @_T[out(A, M, B) \Rightarrow_B rec_T(A, M)] \\
(T4) & \quad @_T[leak \Rightarrow (\bigvee_{B \in Princ} \Box_B [\top])] \\
\end{align*}

and axioms for the interaction of each principal $A \in Princ$ with $T$, that is

\begin{align*}
(P_T1) & \quad @_A[send_T(M, B) \Rightarrow Y(knows(M) \land knows(B))] \\
(P_T2) & \quad @_A[send_T(M, B) \Rightarrow_T in(A, M, B)] \\
(P_T3) & \quad @_A[rec_T(B, M) \Rightarrow_T out(B, M, A)] \\
(P_T4) & \quad @_A[spy_T(B, M) \Rightarrow_T (leak \land P(\bigvee_{C \in Princ} in(B, M, C)))] \\
(P_T6) & \quad @_A[fresh(X) \Rightarrow \neg @_T[\top]].
\end{align*}

An additional axiom is needed to define the state propositions representing evidence for each principal $A \in Princ$, namely

\begin{align*}
(E) & \quad @_A[evid(B, M) \Rightarrow P \ rec_T(B, M)].
\end{align*}

$TTP$ models $\mu$ are those interpretation structures over $\Sigma_{TTP}$ that satisfy the above axioms and the following clause $(K_T)$, which replaces $(K)$ of the CB model. For each $A \in Princ$, $M \in Msg$, and non-empty local state $\xi_A$,

\begin{align*}
(K_T) & \quad \mu, \xi_A \models_A knows(M) \text{ iff } M \in \text{close}\{M' \mid \mu, \xi_A \models (Y \text{ knows}(M')) \lor \text{rec}(M') \lor \text{spy}(M') \lor \text{fresh}(M') \lor (\bigvee_{B \in Princ} \text{rec}_T(B, M') \lor \text{spy}_T(B, M'))\}.
\end{align*}

Clearly, (K1–K7) also follow from $K_T$, as well as the following properties:

\begin{align*}
@_A[rec_T(B, M) \Rightarrow knows(M)] & \quad (K_T5) \\
@_A[spy_T(B, M) \Rightarrow knows(M)] & \quad (K_T6)
\end{align*}

We also strengthen the honesty requirement. Besides axiom $(Hon)$, we require for every $A \in Hon$:

\begin{align*}
(Hon_T) & \quad @_A[\neg \text{spy}_T(B, M)], \text{ for every } B \in Princ \text{ and message } M.
\end{align*}

In $TTP$, we can prove that the state propositions that provide each principal evidence of origin of the messages he received via $T$ are actually correct.

**Proposition 5.1** $TTP \models @_A[evid(B, M)] \Rightarrow @_B[P_0 \ send_T(M, A)]$ for $A, B \in Princ$ and $M \in Msg$.

**Proof:** Follows easily from axioms $(E)$, $(P_T3)$, $(T2)$, and $(T1)$. \qed
5.1.2 Digital signatures

The $DS$ model specializes the channel-based model $CB$ of Section 3. We require that every principal $A$ possesses a secret, special-purpose asymmetric key $K_A^{-1}$, subject to the key axioms (aKey1–aKey2). We assume that these keys are new, i.e., $\{K_A \mid A \in \text{Princ}\} \cap \text{Num} = \emptyset$, and that the network signature $\langle \text{Princ}, \text{Num}\rangle$ is augmented to $\langle \text{Princ}, \text{Num}^+\rangle$, with $\text{Num}^+ = \text{Nonces} \cup \text{Sym}K \cup \text{Pub}K^+$ where $\text{Pub}K^+ = \text{Pub}K \cup \{K_A \mid A \in \text{Princ}\}$. We write $\text{Msg}^+$ to denote the set of messages in the augmented signature, in contrast to $\text{Msg}$, the messages in the original signature. For the purpose of message-origin authentication, we specify that a principal $A$ should indicate the origin of a message $M$ by sending it along with $A$’s name and a signature, that is, $A$ sends $M; A; \{[M]_1^a\}_{K_A^{-1}}$. By using the associated public key $K_A$, the message’s receiver can then verify the signature to determine whether $M$ originates from $A$.

We define $\Sigma_{DS}$ to be identical to $\Sigma_{CB}$, except defined over the augmented network signature. We let $L_{DS}$ denote the DTL language over the signature $\Sigma_{DS}$. To guarantee the desired behavior, we require that each honest principal $A$ only uses his private key for signing messages where signatures or their associated public/private keys do not occur. Similarly, we require that $A$ never receives messages that use the special-purpose public/private keys, unless they are properly signed. Namely, we require

$$\text{(NS)} \quad @A[\neg \text{send}(M', B)] \text{ if } M' \in \text{Msg}^+ \setminus \text{Msg} \text{ and } M' \neq M; A; \{[M]_1^a\}_{K_A^{-1}} \text{ for some } M \in \text{Msg},$$

$$\text{(NR)} \quad @A[\neg \text{rec}(M')] \text{ if } M' \in \text{Msg}^+ \setminus \text{Msg} \text{ and } M' \neq M; B; \{[M]_2^a\}_{K_R^{-1}} \text{ for some } M \in \text{Msg} \text{ and } B \in \text{Princ}. $$

Note that $A \in \text{Hon}$ will never disclose his special-purpose private key $K_A^{-1}$ or forward messages signed by other principals using the special-purpose keys. As above, we assume for simplicity that principals know each other’s names (N) and public keys (PK), and that honest principals do not spy (Hon). The $DS$ models are the interpretation structures that satisfy these axioms along with the $CB$ requirements.

5.1.3 Comparing the models

The $TTP$ model is more abstract than the $DS$ model in that it provides a higher-level message-origin authentication mechanism without using cryptography. Still, we would like that the two models provide message-origin authentication mechanisms with comparable behavior, as depicted in Figure 6, where the triple arrow indicates that $\text{evid}(A, M)$ holds from that point on.

It is clear that we must use the more abstract language of the $TTP$ and provide a translation to the language of the $DS$ model. Given a formula $\gamma \in L_{TTP}$, let $\bar{\gamma} \in L_{DS}$ be the formula obtained from $\gamma$ by uniformly replacing each occurrence of

- $T$ with $Ch$,
- $\text{send}(M, B)$ local to principal $A$ with $\text{send}(M; A; \{[M]_1^a\}_{K_A^{-1}}, B)$,
- $\text{rec}(B, M)$ with $\text{rec}(M; B; \{[M]_2^a\}_{K_B^{-1}})$,
- $\text{spy}(B, M)$ with $\text{spy}(M; B; \{[M]_2^a\}_{K_B^{-1}})$,
- $\text{evid}(B, M)$ with $\text{P}_\mu \text{rec}(M; B; \{[M]_2^a\}_{K_B^{-1}})$,
- $\text{in}(A, M, B)$ local to $T$ with $\text{in}(A, M; A; \{[M]_1^a\}_{K_A^{-1}}, B)$,
- $\text{out}(A, M, B)$ local to $T$ with $\text{out}(A, M; A; \{[M]_1^a\}_{K_A^{-1}}, B)$.

We would like to prove that $TTP$ and $DS$ are equivalent in the sense that $TTP \models \gamma$ if $DS \models \bar{\gamma}$. One way to establish such an equivalence is to define model transformations $\beta : TTP_\Xi \rightarrow DS_\Xi$ and $\theta : DS_\Xi \rightarrow TTP_\Xi$ such that, for every $\gamma \in L_{TTP}$, $(\mu_{TTP}, \xi_{TTP}) \in TTP_\Xi$, and $(\mu_{DS}, \xi_{DS}) \in DS_\Xi$:
\[ TTP \text{ model:} \]

\[ \begin{array}{c}
A & \ldots & \overset{send_T(M, B)}{\rightarrow} & \ldots \\
T & \ldots & \overset{\text{in}(A, M, B)}{\rightarrow} & \overset{\text{evid}(A, M)}{\rightarrow} & \overset{\text{out}(A, M, B)}{\rightarrow} & \ldots \\
B & \ldots & \overset{\text{rec}_T(A, M)}{\rightarrow} & \ldots
\end{array} \]

\[ DS \text{ model:} \]

\[ \begin{array}{c}
A & \ldots & \overset{send(M; A; \{\|M\|_{K_a}^{A-1}\}, B)}{\rightarrow} & \ldots \\
Ch & \ldots & \overset{\text{in}(A, M; A; \|M\|_{K_a}^{A-1}, B)}{\rightarrow} & \overset{\text{out}(A, M; A; \|M\|_{K_a}^{A-1}, B)}{\rightarrow} & \ldots \\
B & \ldots & \overset{\text{rec}(M; A; \|M\|_{K_a}^{A-1})}{\rightarrow} & \ldots
\end{array} \]

Figure 6: Comparing a TTP model and a DS model.

(i) \( \mu_{TTP}, \xi_{TTP} \models \gamma \) if and only if \( \beta(\mu_{TTP}, \xi_{TTP}) \models \tau \), and

(ii) \( \theta(\mu_{DS}, \xi_{DS}) \models \gamma \) if and only if \( \mu_{DS}, \xi_{DS} \models \tau \).

Observe that the composition of the two model transformations \( \beta \) and \( \theta \) need not be the identity, but only preserve logical equivalence for \( L_{TTP} \) modulo translation of formulas to the language of the DS model.

There are two subtle issues here. First, the model transformations, associated to \( \beta \) and guided by the syntactic translation defined above, must merge \( Ch \) and \( T \) together, since there is only one communication medium in DS\(^4\). Hence, one cannot expect property (i) to hold in general when \( \gamma \) involves temporal formulas local to either the public or the logged channel. This is, however, a minor restriction, as we should still be able to prove (i) for all relevant properties concerning the behavior of the principals.

The second issue concerns property (ii): the transformation \( \theta \) must be able to represent in TTP all behaviors allowed in DS models. The problem is that there is a minor incompatibility between the models related to the use of the special-purpose keys in DS. The axioms (NS) and (NR) require that honest agents use signatures appropriately in their realization of a trusted channel. However, a priori, it does not seem reasonable to similarly restrict the intruder. Still, it is clear that \( Z \) gains nothing from sending messages that no other principal is willing to receive. Hence, as a consequence of axiom (EZ1), we have the following property:

(\( NS_Z \)) \( @_{Z}[\neg \text{send}(M', A)] \) if \( M' \in \text{Msg}^+ \setminus \text{Msg} \) and \( M' \neq M; B; \|M\|_{K_b}^{a-1} \) for some \( M \in \text{Msg} \) and \( B \in \text{Princ} \).

(\( NS_Z \)) is weaker than the axiom (NS) for the honest principals. As a consequence, \( Z \) can still prevent message-origin authentication in the DS model, which we established for the TTP model in Proposition 5.1. The reason is that in the TTP model, \( T \) not only issues evidence of message origin but it also provides a stronger form of correspondence, in that the intended recipient indeed receives the signed message. However, this cannot be guaranteed in the DS model since nothing prevents \( Z \) from spying a message \( M; A; \|M\|_{K_a}^{a} \) sent by a principal \( A \) to a principal \( B \), and then forwarding it to some other principal \( C \). Clearly, when receiving it, \( C \) will have evidence of the message’s origin but cannot be sure that \( A \) really sent the message to him. This situation, which

\(^4\)We will refrain here from working out the details of this model transformation since we will do this in detail shortly in a slightly different, but compatible, context.
Figure 7: Z interferes with message-origin authentication in the DS model

is illustrated in Figure 7, cannot be mimicked in the TTP model. Thus, there are two options for establishing the desired equivalence:

1. Relax the TTP model to allow the intruder to divert messages from T and forward them to different destinations.

2. Change the way messages are signed in the DS model to guarantee the required form of correspondence.

Although the second option is more appropriate (as discussed in [51]), both options have advantages and disadvantages, which we discuss below.

Option 1. We can solve the above mismatch by changing the TTP model to allow the situation described in Figure 7. Namely, we consider a slightly different model $TTP'$ where the intruder can divert each message sent to the logged channel $T$ to a different recipient. To formalize this, we extend the TTP signature with an additional action $dvt_T$ for each principal and we change or add new axioms to specify its properties. The signature $\Sigma_{TTP'} = (Princ \uplus \{Ch, T\}, Act, Prop)$ is such that for each $A \in Princ$, we have that

- $Act_A$ is composed of
  
  $send(M, B), \, rec(M), \, spy(M), \, fresh(X), \, send_T(M, B), \, rec_T(B, M), \, and \, spy_T(B, M)$,
  
  as in $\Sigma_{TTP}$, and
  
  $dvt_T(B, M, C)$: divert to principal $C$ the message $M$ originating from $B$;

- $Prop_A$ includes, as in $\Sigma_{TTP}$, the state propositions $knows(M)$ and $evid(B, M)$;

and for $Ch$ and $T$, as in $\Sigma_{TTP}$, we have that

- $Act_{Ch} = Act_T$ includes the actions $in(A, M, B), \, out(A, M, B)$, and $leak$;

- $Prop_{Ch} = Prop_T = \emptyset$;

We use $L_{TTP'}$ to denote the DTL language over the extended signature $\Sigma_{TTP'}$. We specify the new divert actions by replacing axiom $(T1)$ with

$$(T1')@_{T}[in(B, M, C) \Rightarrow (\bigcirc_B[send_T(M, C)] \lor (\forall_{A \in Princ} \bigcirc_A[dvt_T(B, M, C)]))]$$

and adding additional axioms for each principal $A$, namely:

$$(P_7)@_{A}[dvt_T(B, M, C) \Rightarrow P\, spy_T(B, M)]$$

$$(P_8)@_{A}[dvt_T(B, M, C) \Rightarrow_T in(B, M, C)].$$

28
Note that as a consequence of axioms \((\text{Hon}_T)\) and \((\text{P}_T 7)\), honest principals are also not allowed to divert messages. The \(TTP'\) models are then the interpretation structures over \(\Sigma_{TTP'}\) satisfying \((\text{F}1 - \text{F}2), (\text{C}1 - \text{C}4), (\text{P}1 - \text{P}6), (\text{T}1'), (\text{T}2 - \text{T}4), (\text{P}_T 1 - \text{P}_T 4), (\text{P}_T 6 - \text{P}_T 8), (\text{E}), (\text{N})\), and \((\text{K}_T)\). In this setting, we can no longer interpret message-origin authentication as we did in \(TTP\) and Proposition 5.1 is weakened as follows.

**Proposition 5.2** \(TTP' \models \Diamond_{A}[\text{evid}(B, M)] \Rightarrow \Diamond_{B}[\text{send}_T(M, C)]\) for \(A, B \in \text{Princ}\) and \(M \in \text{Msg}\).

*Proof:* This follows from the definition of satisfaction, using the axioms \((\text{E}), (\text{P}_T 3), (\text{T}2), (\text{T}1'), (\text{P}_T 7),\) \((\text{P}_T 4)\) and the fact that interpretation structures are bounded to the past. \(\square\)

The models \(TTP'\) and \(DS\) can now be shown to be equivalent, in the precise sense described above, where we extend the translation of formulas \(\gamma \in \mathcal{L}_{TTP'}\) to \(\overline{\gamma} \in \mathcal{L}_{DS}\) by additionally replacing each occurrence of

- \(\text{dvt}_T(B, M, C)\) with \(\text{send}(M; B; [M]^{a}_{K_B^{-1}}, C)\).

**Proposition 5.3** For every \(\gamma \in \mathcal{L}_{TTP'}^D\) built from formulas private to principals, \(TTP' \models \overline{\gamma}\) if and only if \(DS \models \gamma\).

We omit the proof as it is technically very similar to the upcoming proof of the corresponding property that we give for the second option. As a corollary, we have that the model \(DS\) based on digital signatures fulfills a form of message-origin authentication similar to the one stated in Proposition 5.2 for \(TTP'\), where evidence that a message \(M\) originated in \(B\) is provided to a principal \(A\) by the past reception of \(M; B; [M]^{a}_{K_B^{-1}}\).

**Corollary 5.4** \(DS \models \Diamond_{A}[\text{rec}(M; B; [M]^{a}_{K_B^{-1}})] \Rightarrow \Diamond_{B}[\text{send}(M; B; [M]^{a}_{K_B^{-1}}, C)]\) for \(A, B \in \text{Princ}\) and \(M \in \text{Msg}\).

**Option 2.** In this option, we keep the initial formulation of the \(TTP\) model, as well as the original meaning of message-origin authentication, but add more structure to the digitally-signed messages in the concrete model. We let \(\Sigma_{DS} = \Sigma_{DS}\), but we will still use \(\mathcal{L}_{DS}\) to denote the corresponding DTL language. We change \(DS^F\) by requiring that a principal \(A\) must now send a message \(M\) whose message-origin authentication is required by the recipient \(B\) including the name \(B\) as part of the signed message, that is, \(A\) sends \(M; A; [B; M]^{a}_{K_A^{-1}}\) (cf. [51]). In addition, axioms \((\text{NS})\) and \((\text{NR})\) must be rewritten, for every honest \(A\), to:

\[\text{\text{(NS)} } \neg \text{send}(M', B), \text{ if } M' \in \text{Msg} \setminus \text{Msg} \text{ and } M' \neq M; A; [B; M]^{a}_{K_A^{-1}}, \text{ for some } M \in \text{Msg} \text{ and } B \in \text{Princ}\]

\[\text{\text{(NR)} } \neg \text{rec}(M'), \text{ if } M' \in \text{Msg} \setminus \text{Msg} \text{ and } M' \neq M; B; [A; M]^{a}_{K_B^{-1}}, \text{ for some } M \in \text{Msg} \text{ and } B \in \text{Princ}\]

Using axiom \((\text{EZ}1)\), we also obtain a variant of the \((\text{NS}_2)\) property, namely

\[\text{\text{(NS)} } \neg \text{send}(M', A), \text{ if } M' \in \text{Msg} \setminus \text{Msg} \text{ and } M' \neq M; B; [A; M]^{a}_{K_B^{-1}}, \text{ for some } M \in \text{Msg} \text{ and } B \in \text{Princ}.\]

Finally, we redefine the translation between the languages of the two models. Given \(\gamma \in \mathcal{L}_{TTP}\), let \(\overline{\gamma}\) be the formula obtained from \(\gamma\) by uniformly replacing each occurrence of

- \(T\) with \(\text{Ch}\),
- \(\text{send}_T(M, B)\) local to principal \(A\) with \(\text{send}(M; A; [B; M]^{a}_{K_A^{-1}}, B)\),
principals, it suffices to show that a formula is private to principals. As any such formula is only if \( DS \) holds in the model. The result follows by a trivial adaptation of Lemma 2.2, once we note that the translation of formulas matches the relabeling of the events introduced by \( \beta \) (and \( \theta \)), and that \( \pi \) and \( \pi^g \) agree for messages in \( Msg \).

\[ \alpha_A(\epsilon) = \begin{cases} \alpha_A(\epsilon) & \text{if } \alpha_A(\epsilon) \text{ is an } A\text{-action in } \Sigma_{DS} \\
      \text{send}(M; A; |B; M|_{K_A^{-1}}^a, B) & \text{if } \alpha_A(\epsilon) = \text{send}_T(M, B) \\
      \text{rec}(M; B; |A; M|_{K_B^a}) & \text{if } \alpha_A(\epsilon) = \text{rec}_T(B, M) \\
      \text{spy}(M; B; |C; M|_{K_A^a}) & \text{if } \alpha_A(\epsilon) = \text{spy}_T(B, M) \text{ and } \\
      \alpha_T(\epsilon') = \text{in}(B, M, C), \text{ for some } \epsilon' < T \epsilon \\
\end{cases} \]

\[ \alpha_C^a(e) = \begin{cases} \alpha_C(\epsilon) & \text{if } e \in Ev_{Ch} \\
      \text{in}(A, M; A; |B; M|_{K_A^{-1}}^a, B) & \text{if } e \in Ev_T \text{ and } \alpha_T(\epsilon) = \text{in}(A, M, B) \\
      \text{out}(A, M; A; |B; M|_{K_A^{-1}}^a, B) & \text{if } e \in Ev_T \text{ and } \alpha_T(\epsilon) = \text{out}(A, M, B) \\
      \text{leak} & \text{if } e \in Ev_T \text{ and } \alpha_T(\epsilon) = \text{leak} \\
\end{cases} \]

\[ \pi_A^g(\emptyset) = \{ \text{knows}(M') \mid |M' | = \text{close}(\{ M \mid \text{knows}(M) \in \pi_A(\emptyset) \}) \cup \{ K_B | B \in \text{Princ} \cup \{ K_A^{-1} \} \} \}, \]

whereas \( \pi_{Ch}^g \) is always empty.

Note that when \( \alpha_A(\epsilon) = \text{spy}_T(B, M) \), we assume fixed a consistent choice of principal \( C \) such that \( \alpha_A^a(\epsilon) = \text{spy}_T(M; B; |C; M|_{K_A^a}) \), as guaranteed by axiom (P_T4). In addition, as in all other cases, when seen as a formula, \( \alpha_A^a(\epsilon) \) corresponds to \( \alpha_A(\epsilon)^g \). Note also that \( \alpha_C^a \) is well-defined since \( Ev_{Ch} \cap Ev_T = \emptyset \). Moreover, \( \text{knows}(M) \in \pi_A^g(\emptyset) \) iff \( \text{knows}(M) \in \pi_A(\emptyset) \) for every \( M \in Msg \). Whenever one defines a model transformation such as this one, one must guarantee that indeed \( \langle \lambda^g, \alpha^g, \pi^g \rangle \in DS^g \). It is straightforward, though tedious, to check that all the necessary conditions are satisfied by the interpretation structure. For instance, note that axiom (NS^g) holds by construction as the only sending actions of a principal \( A \) are either normal channel-sending actions like \( \text{send}(M, B) \) with \( M \in Msg \), or else they come from actions like \( \text{send}_T(M, B) \) where again \( M \in Msg \) and they are transformed into \( M; A; |B; M|_{K_A^{-1}}^a \). Another axiom, such as (C1), holds in \( DS^g \) as a consequence of (C1) and (T1) holding in any TTP model. Observe, moreover that, by construction, \( \xi \) is still a global state of \( \langle \lambda^g, \alpha^g, \pi^g \rangle \).

We must prove that \( \mu_{TTP}, \xi \models \gamma \) iff \( \beta(\mu_{TTP}, \xi) \models \gamma^g \), for every \( \gamma \in L^g_{TTP} \) built from formulas private to principals. As any such \( \gamma \) must be a Boolean combination of private formulas of different principals, it suffices to show that \( \mu_{TTP}, \xi \models \emptyset \) iff \( \beta(\mu_{TTP}, \xi) \models \emptyset^g \), for every \( A \in \text{Princ} \) and \( \varphi \in L^g_A \). The result follows by a trivial adaptation of Lemma 2.2, once we note that the translation of formulas matches the relabeling of the events introduced by \( \beta \) (and \( \theta \)), and that \( \pi \) and \( \pi^g \) agree for messages in \( Msg \).
Conversely, given a $DS$ model $\mu_{DS} = \langle \lambda^2, \alpha^2, \pi^z \rangle$, we must be able to identify the channel events that correspond, in a TTP model, to the trusted third-party $T$. Of course, these are precisely the events whose label is somehow related to a message that uses the special-purpose signatures. Below, we will call $e \in Ev^T_{\lambda}$ a $T$-event if one of the following conditions holds:

- $\alpha^T_{\lambda}(e) = in(A, M, B)$ or $\alpha^T_{\lambda}(e) = out(A, M, B)$, for some $A, B \in Princ$ and some $M \in Msg^+ \setminus Msg$; or
- $\alpha^T_{\lambda}(e) = leak$, and $e \in Ev^T_{Z}$ with $\alpha^T_{Z}(e) = spy(M)$, for some $M \in Msg^+ \setminus Msg$.

Note that, as a consequence of conditions $(NS^2)$, $(NR^2)$, and $(NS^2_5)$, all the relevant messages $M \in Msg^+ \setminus Msg$ appearing in the labels of the events above must be of the form $M': C; [D; M']^a_K\Sigma$, for some $C, D \in Princ$ and $M' \in Msg$. Furthermore, in the context above, condition $(EZ2)$ also ensures that $A = C$ and $B = D$. We now let $\theta : DS^S_\Sigma \rightarrow TTP_\Sigma$ be defined for each pair $\langle \mu_{DS}, \xi \rangle \in DS^S_\Sigma$ by $\theta(\mu_{DS}, \xi) = \langle \lambda, \alpha, \pi, \xi \rangle$ where

- $\lambda_A = \lambda^2_A$ for every $A \in Princ$, $\lambda_{\lambda \lambda} = \langle Ev_{\lambda \lambda}, \leq_{\lambda \lambda} \rangle$, and $\lambda_T = \langle Ev_T, \leq_T \rangle$, where $Ev_T = \{ e \in Ev^T_{\lambda} \mid e \text{ is a } T\text{-event}, Ev_{\lambda \lambda} = Ev^T_{\lambda \lambda} \setminus Ev_T, \text{ and } \leq_{\lambda \lambda} \text{ and } \leq_T \text{ are the corresponding restrictions of } \leq_{\lambda \lambda} \}$;
- $\alpha_A(e) = \begin{cases} \alpha^T_{\lambda}(e) \text{ if } \alpha^T_{\lambda}(e) \text{ does not involve a message in } Msg^+ \setminus Msg \\ send_T(B, M) \text{ if } \alpha^T_{\lambda}(e) = send(M; A; \Sigma; B; M^a_K\Sigma) \\ recy(B, M) \text{ if } \alpha^T_{\lambda}(e) = recy(M; B; M^a_K\Sigma) \\ spy_T(B, M) \text{ if } \alpha^T_{\lambda}(e) = spy(M; B; M^a_K\Sigma), \end{cases}$
- $\alpha_{\lambda \lambda}(e) = \alpha^T_{\lambda \lambda}(e)$ for $e \in Ev_{\lambda \lambda}$, and
- $\alpha_T(e) = \begin{cases} in(A, M, B) \text{ if } \alpha^T_{\lambda}(e) = in(A, M; B; M^a_K\Sigma) \\ out(A, M, B) \text{ if } \alpha^T_{\lambda}(e) = out(A, M; B; M^a_K\Sigma) \\ leak \text{ if } \alpha^T_{\lambda}(e) = leak; \end{cases}$
- $\pi_\theta(\emptyset) = \{ knows(M) \mid M \in \pi^z_A(\emptyset) \cap Msg \}$, and $\pi_{\lambda \lambda}$ and $\pi_T$ are both always empty.

It is tedious although not difficult to prove that $\langle \lambda, \alpha, \pi, \xi \rangle \in TTP$ and that $\xi$ is still a global state. Moreover, by construction, we have that $\beta$ and $\theta$ are almost inverses of each other, in the sense that $\langle \mu_{DS}, \xi \rangle$ and $\beta(\theta(\mu_{DS}, \xi))$ may differ only in the choice of a linearization compatible with $\xi$ in the construction of the latter that is different from the linearization implicit in the former. Still, the two models are similar enough to satisfy the preconditions of Lemma 2.2, again, with respect to all formulas $\gamma \in L^T_{\Theta}$, built from formulas private to principals. As a consequence, we also obtain that $\mu_{DS}, \xi \models \tau$ if and only if $\beta(\theta(\mu_{DS}, \xi)) \models \tau$ if $\theta(\mu_{DS}, \xi) \models \gamma$.

Finally, we can establish the equivalence of the two models. If $TTP \models \gamma$ and $\langle \mu_{DS}, \xi \rangle \in DS^S_\Sigma$ then $\theta(\mu_{DS}, \xi) \in TTP_\Sigma$ and thus $\theta(\mu_{DS}, \xi) \models \gamma$. Hence, $\mu_{DS}, \xi \models \tau$ and thus $DS^S \models \tau$. Conversely, if $DS^S \models \tau$ and $\langle \mu_{TTP}, \xi \rangle \in TTP_\Sigma$ then $\beta(\mu_{TTP}, \xi) \models \tau$ and so $\beta(\mu_{TTP}, \xi) \models \tau$. Hence, $\mu_{TTP}, \xi \models \gamma$ and thus $TTP \models \gamma$. \hfill \Box

Figure 8 depicts the essential ingredients of the proof of Proposition 5.5 and it closely follows the translation of syntax. Note that the upper and lower diagrams depict two $DS^S$ models that differ only in the linearization of the $\lambda$ and $T$ events of the diagram in the middle. Note further that $\beta$ transforms the middle diagram along with the global state $\xi$, but the linearization considered would not be compatible with the global state $\xi'$ also depicted.

As a corollary of Proposition 5.5, we have that $DS^S$ fulfills the message-origin authentication requirement originally stated for $TTP$ in Proposition 5.2.
is transformed by $\theta$ into

which is transformed back by $\beta$ into

Figure 8: Model transformations between TTP and $DS^\xi$ models.

Corollary 5.6 $DS^\xi \models \otimes_A [P_0 \text{rec}(M; B; [A; M]^{a}_{K_B^{-1}})] \Rightarrow \otimes_B [P_0 \text{send}(M; B; [A; M]^{a}_{K_B^{-1}}, A)]$ for $A, B \in \text{Princ}$ and $M \in \text{Msg}$.

5.2 Channel-based versus intruder-centric models

We now show that, from the perspective of protocol analysis, the channel-based models we have been using are equivalent to the kinds of intruder-centric models used in many protocol analysis tools [3, 12, 59, 65]. In intruder-centric models, the intruder controls, and is identified with, the network. Moreover, it is even possible to eliminate the honest agents. This avoids the complications of distribution as intruder-centric models correspond to linear models of the network behavior.

Specifically, we will establish a notion called attack equivalence between models, which says that attacks are neither lost nor gained by moving between models, where an attack is a countermodel to a security property. In this paper, we restrict our focus to secrecy and authentication properties.
as described in Section 3.4.

We will proceed in three phases. In the first phase, we translate our CB models to models where the intruder is merged with the channel, which we call ZB models. Hence, ZB models no longer have a channel and messages sent go directly to the intruder. In the second phase, which we call step compression, we reorganize the ZB models so that the actions of each principal corresponding to the execution of certain protocol steps appear as consecutive events in local life-cycles. We call these CZB models, standing for compressed ZB models. In the third and final phase, we drop any explicit reference to the honest principals in CZB models, keeping just the intruder. This requires changing the granularity of the actions. These are no longer the atomic actions of sending and receiving messages or generating fresh data, but rather transactions corresponding to the execution of complete steps of some fixed protocol by some agent. We call the resulting intruder-centric models ZC models.

Below, we introduce all the models and show their attack equivalence by proving properties of the proposed model transformations. Before doing so, we take a closer look at protocols and how we model them. Recall from Section 3 that an Alice-and-Bob-style protocol description corresponds to a sequence of message exchanges

\[(msg_1) \quad s_1 \rightarrow r_1 \quad : \quad (f_{1,1}, \ldots, f_{1,t_1}). \quad M_1 \]

\[
\vdots
\]

\[(msg_q) \quad s_q \rightarrow r_q \quad : \quad (f_{q,1}, \ldots, f_{q,t_q}). \quad M_q
\]

\[(msg_{q+1}) \quad s_{q+1} \rightarrow r_{q+1} \quad : \quad (f_{q+1,1}, \ldots, f_{q+1,t_{q+1}}). \quad M_{q+1}
\]

\[
\vdots
\]

\[(msg_m) \quad s_m \rightarrow r_m \quad : \quad (f_{m,1}, \ldots, f_{m,t_m}). \quad M_m
\]

where each \( s_i, r_i \in \{a_1, \ldots, a_j\} \) identifies the principal playing one of the \( j \) protocol roles. We assume, as is standard and without loss of generality, that \( s_{j+1} = r_j \), for \( 1 \leq j < m \).

Whereas we previously considered the protocol run corresponding to each principal given a specific protocol instantiation, we now split the runs in smaller steps. Specifically, consider two consecutive lines of the protocol description, e.g., those labeled \( msg_q \) and \( msg_{q+1} \). Then there is a sequence of actions that the principal instantiating \( r_q = s_{q+1} \) must execute. Namely, he must receive \( M_q \), freshly generate \( f_{q+1,1}, \ldots, f_{q+1,t_{q+1}} \), and finally send \( M_{q+1} \). In general, such a protocol description gives rise to \( m + 1 \) protocol steps, where the first protocol step (by the principal instantiating \( s_1 \)) does not include an initial receiving action and the last protocol step (by the principal instantiating \( s_m \)) consists only of receiving the last message.

Formally, given a protocol instantiation \( \sigma \), a protocol role \( 1 \leq i \leq j \), and \( 0 \leq k \leq m \), we define \( step_k^i = msg_k^i \cdot msg_{k+1}^i \), where we assume that \( msg_0^i = msg_{m+1}^i = () \). Thus, we have:

\[
\text{step}_k^i = \begin{cases} 
(fresh(f_{1,1}) \ldots fresh(f_{1,t_1}).send(M_1, r_1)) & \text{if } k = 0 \text{ and } s_1 = a_i \\
(rec(M_k).fresh(f_{k+1,1}) \ldots fresh(f_{k+1,t_{k+1}}).send(M_{k+1}, r_{k+1})) & \text{if } 0 < k < m \text{ and } r_k = s_{k+1} = a_i \\
(rec(M_m)) & \text{if } k = m \text{ and } r_m = a_i \\
() & \text{otherwise}
\end{cases}
\]

In this way, \( run^i = msg_1^i \cdot \ldots \cdot msg_m^i = step_0^i \cdot \ldots \cdot step_m^i \). If \( \sigma(a_i) = A \) then we write \( step_{A,k}^i(\sigma) = \sigma(step_k^i) \), or simply \( step_{A,k}^i(\{\pi\}, \{\tau\}) \) where \( \pi \) and \( \tau \) stand, respectively, for the sequences of identifiers and fresh data symbols used in the protocol description.

Below, in step-compressed models, we will require that the events whose labels match a given protocol step by a principal must be consecutive, that is, protocol steps will not be interleaved with other actions. Then, in intruder-centric models, we will introduce explicit (trans)actions \( trans_{A,k}^i(\sigma) \) to represent each protocol step \( step_{A,k}^i(\sigma) \). In Figure 9, we illustrate the successive transformation of a CB model to a ZB, CZB, and ZC model. For simplicity, we consider there a
The CB model

\[ \begin{align*}
\text{A} & \quad \text{fresh}(F_1) \quad \text{send}(M_1, B) \\
\text{B} & \quad \text{rec}(M_1) \quad \text{fresh}(F_2) \quad \text{fresh}(F'_1) \quad \text{send}(M_2, A) \quad \text{send}(M'_1, C) \\
\text{Ch} & \quad \text{in}(A, M_1, B) \quad \text{out}(A, M_1, B) \\
\text{Z} & \quad \text{fresh}(F'_2) \quad \text{spy}(M_2) \quad \text{send}(M'_2, A) \\
\end{align*} \]

is attack equivalent (by Corollary 5.13) to the ZB model

\[ \begin{align*}
\text{A} & \quad \text{fresh}(F_1) \quad \text{send}(M_1, B) \\
\text{B} & \quad \text{rec}(M_1) \quad \text{fresh}(F_2) \quad \text{fresh}(F'_1) \quad \text{send}(M_2, A) \quad \text{send}(M'_1, C) \\
\text{Z} & \quad \text{in}(A, M_1, B) \quad \text{out}(Z, M_1, B) \\
\end{align*} \]

which is attack equivalent (by Corollary 5.17) to the CZB model

\[ \begin{align*}
\text{A} & \quad \text{fresh}(F_1) \quad \text{send}(M_1, B) \\
\text{B} & \quad \text{rec}(M_1) \quad \text{fresh}(F_2) \quad \text{fresh}(F'_1) \quad \text{send}(M_2, A) \quad \text{send}(M'_1, C) \\
\text{Z} & \quad \text{in}(A, M_1, B) \quad \text{out}(Z, M_1, B) \\
\end{align*} \]

which is attack equivalent (by Corollary 5.30) to the ZC model

\[ \begin{align*}
\text{Z} & \quad \text{trans}_{\text{in}}^{\text{rec}}(A, B, F_1) \quad \text{fresh}(F'_2) \quad \text{trans}_{\text{in}}^{\text{rec}}(A, B, F_1, F'_2) \quad \text{trans}_{\text{in}}^{\text{rec}}(B, C, F'_1) \quad \text{trans}_{\text{in}}^{\text{rec}}(A, B, F_1, F'_2) \\
\end{align*} \]

Figure 9: From CB to ZC models (via ZB and CZB models)

protocol of the form

\[
\begin{align*}
a \to b : & \quad (f_1) \cdot M_1 \\
b \to a : & \quad (f_2) \cdot M_2
\end{align*}
\]

The figure depicts a (possible attack) scenario where honest principals A and B are running the protocol, but B’s reply in the second message is intercepted by the intruder, who then sends a fake message to A. While this happens, B is also starting another protocol run with C.

5.2.1 Phase 1: Intruder as the channel

The first phase of our reduction is to merge the intruder with the channel, as illustrated by the two topmost models in Figure 9. We call the resulting models intruder-based models, or ZB models for short. In this step, we keep the same network signature (Princ, Num). The intruder-based signature \( \Sigma_{\text{ZB}} \) is a distributed signature obtained from the network signature, analogously to how we obtained the channel-based signature in Section 3.2. However, we now drop the channel, replacing it by the intruder, and the actions send and rec of the intruder become the actions in and out of the new intruder. Moreover, when constructing a ZB model from a CB model, we remove all events representing interaction between the channel and the intruder. Hence, we delete
the actions spy, send, and rec of the intruder and the action leak of the channel. The signature
\( \Sigma_{ZB} = \langle \text{Princ}, \text{Act}, \text{Prop} \rangle \) is such that

- for each \( A \in \text{Hon} \), \( \text{Act}_A \) is composed of \( \text{send}(M, B) \), \( \text{rec}(M) \), and \( \text{fresh}(X) \), and \( \text{Prop}_A \) is composed of the state propositions \( \text{knows}(M) \), where \( B \in \text{Princ}, M \in \text{Msg} \), and \( X \in \text{Nonces} \uplus \text{SymK} \uplus \text{PubK} \);

- \( \text{Act}_Z \) contains \( \text{in}(A, M, B) \), \( \text{out}(A, M, B) \), and \( \text{fresh}(X) \), and \( \text{Prop}_Z \) contains the state propositions \( \text{knows}(M) \), where \( A, B \in \text{Princ}, M \in \text{Msg} \), and \( X \in \text{Nonces} \uplus \text{SymK} \uplus \text{PubK} \).

The \( ZB \) models \( \mu \) are those satisfying the axioms \( (K) \), for honest principals, \( (F1–F2) \) of the \( CB \) model, which guarantee the freshness and uniqueness of nonces, as well as:

\[
(KZ) \quad \mu_Z, \xi_Z \models_Z \text{knows}(M) \text{ iff } M \in \text{close}(\{M' | \mu_Z, \xi_Z \models_Z (Y \text{ knows}(M')) \vee \bigvee_{A,B \in \text{Princ}} \text{in}(A, M, B, B') \}) \text{ for every non-empty local state } \xi_Z
\]

\[
(Z1) \quad @Z[\text{in}(A, M, B) \Rightarrow \Box_A[\text{send}(M, B)]]
\]

\[
(Z2') \quad @Z[\text{out}(Z, M, B) \Rightarrow \gamma(\text{knows}(M) \land \text{knows}(B))]
\]

\[
(Z3) \quad @Z[\text{out}(A, M, B) \Rightarrow \Box_B[\text{rec}(M)]]
\]

\[
(P1) \quad @A[\text{send}(M, B) \Rightarrow \gamma(\text{knows}(M) \land \text{knows}(B))], \text{ for every } A \in \text{Hon}
\]

\[
(P2) \quad @A[\text{send}(M, B) \Rightarrow @Z[\text{in}(A, M, B)]]], \text{ for every } A \in \text{Hon}
\]

\[
(P3) \quad @A[\text{rec}(M) \models_Z \text{out}(Z, M, A)], \text{ for every } A \in \text{Hon}
\]

\[
(P5) \quad @A[\bigwedge_{B \in \text{Hon} \setminus \{A\}} \neg \Box_B[\top]]
\]

\[
(P6') \quad @A[\text{fresh}(X) \Rightarrow \neg \Diamond_Z[\top]]
\]

\[
(P6'') \quad @Z[\text{fresh}(X) \Rightarrow \bigwedge_{A \neq Z} \neg \Box_A[\top]]
\]

Axiom \( (KZ) \) adapts axiom \( (K) \) for the intruder, which handles the new intruder action in, from which \( Z \) can learn new messages. The axioms for the channel are adapted to the new model, replacing the channel by the intruder. Axioms \( (Z1) \) and \( (Z3) \) (corresponding to \( (C1) \) and \( (C3) \), respectively) are not affected, apart from name replacement. Axiom \( (C2) \) is replaced by \( (Z2') \), as we assume that all outputs are generated by the intruder and therefore do not have corresponding input actions. The only requirement, in this case, is that the intruder knows all the necessary information to produce such messages. Axiom \( (C4) \) is simply dropped, as spying actions are no longer needed. Some of the axioms for the principals’ behavior are also adapted. Axiom \( (P1) \) is as before but with the restriction to honest agents, which applies also to some of the other axioms. In axioms \( (P2) \) and \( (P3) \), the channel is replaced by the intruder. Axiom \( (P4) \) is dropped as there are no longer spy and leak actions. Axiom \( (P5) \) is adapted to allow honest principals to communicate with the intruder but, as before, with no other honest principals. Axiom \( (P6) \) is split into \( (P6') \) and \( (P6'') \), for honest agents and for the intruder, respectively.

As we changed the signature for the intruder by replacing \( \text{send} \) and \( \text{rec} \) actions by \( \text{out} \) and \( \text{in} \) actions, our formulas must be rewritten accordingly. From now on, given a formula \( \gamma \) over a \( CB \) signature where \( \text{spy} \) and \( \text{leak} \) do not occur, we denote by \( \gamma^1 \) the formula over the \( ZB \) signature obtained from \( \gamma \) by uniformly replacing all occurrences of \( \text{send}(A, M) \) local to \( Z \) by \( \text{out}(Z, M, A) \), and all occurrences of \( \text{rec}(M) \) local to \( Z \) by \( \bigvee_{A \in \text{Hon}} \text{in}(A, M, Z) \). In particular, \( \text{secr}_S(\sigma)^1 \) is the same as \( \text{secr}_S(\sigma) \) when all the principals are honest. The situation is similar for authentication, if both principals are honest. Note, however, that \( Z \) now controls the channel and outputs all the messages received by honest principals. Hence, it makes no sense for honest principals to authenticate the intruder.

Our goal is to prove that the two models are attack equivalent. We therefore start by defining a model transformation \( \beta : \text{CB}_Z \rightarrow \text{ZB}_Z \). For each \( \langle \mu_{CB}, \xi \rangle \in \text{CB}_Z \), with \( \mu_{CB} = \langle \lambda, \alpha, \pi \rangle \), let \( \beta(\mu_{CB}, \xi) = \langle \langle \lambda^1, \alpha^1, \pi^1 \rangle, \xi^1 \rangle \) such that
• \( \mu_A^\dagger = \mu_A \) for \( A \in \text{Hon} \) and \( \mu_Z^\dagger \) is such that

\[
\lambda_Z^\dagger = (\text{Ev}_Z^\dagger, \leq_Z^\dagger), \text{ where } \text{Ev}_Z^\dagger = (\text{Ev}_Z \cup \text{Ev}_{CB}) \setminus (\text{Ev}_Z \cap \text{Ev}_{CB}) \text{ and } \leq_Z^\dagger \text{ is some discrete linearization of } \rightarrow_{CB} \cup \rightarrow_Z \text{ restricted to } \text{Ev}_Z^\dagger \text{ that has } (\xi | Z \cup \xi_{\text{CB}}) \cap \text{Ev}_Z^\dagger \text{ as a local state;}
\]

\(- \alpha_Z^\dagger(e) = \begin{cases} 
\alpha_{\text{CB}}(e) & \text{if } e \in \text{Ev}_{\text{CB}} \text{ and } \alpha_{\text{CB}}(e) = \text{in}(B, M, C) \text{ for some } B, M, C \\
\text{out}(Z, M, C) & \text{if } e \in \text{Ev}_{\text{CB}} \text{ and } \alpha_{\text{CB}}(e) = \text{out}(B, M, C) \text{ for some } B, M, C \\
\alpha_Z(e) & \text{if } e \in \text{Ev}_Z \setminus \text{Ev}_{\text{CB}};
\end{cases}
\]

\(- \pi_Z^\dagger \text{ is inductively defined as expected, requiring } \pi_Z^\dagger(\emptyset) = \pi_Z(\emptyset);
\]

• \( \xi_A^\dagger = \xi_A \) and \( \xi_Z^\dagger = (\xi_{\text{CB}} \cup \xi_Z) \cap \text{Ev}_Z^\dagger \).

Note that \( \alpha^\dagger \) is defined so that \( \alpha_Z^\dagger(e) = \alpha_A(e)^\dagger \), for every \( A \in \text{Princ} \) and \( e \in \text{Ev}_A^\dagger \). We claim that \( (\mu^\dagger, \xi^\dagger) \in ZB_Z \). As for the construction of \( \mu^\dagger \), observe that the intruder in \( \mu^\dagger \) is obtained by merging the old intruder with the channel. In doing so, we drop some events, namely all synchronizations between the channel and the (old) intruder, e.g., leak and spy(M2), and send(M3, A) in the CB model in Figure 9. Hence, all that remains from the old intruder are the fresh actions. Moreover, the local successor relation \( \rightarrow_Z^\dagger \) is defined by a discrete linearization of the union of the successor relations of the channel and the old intruder, e.g., \( \text{fresh}(F_3) \) is moved before \( \text{out}(Z, M_3, A) \) in the ZB model in Figure 9. Note that it is always possible to linearize these two relations in a way compatible with an existing state \( \xi \). Note also that, as a consequence of the remarks above, the map \( \beta \) is not injective.

We now prove some properties of this model transformation.

**Lemma 5.7** Let \( \langle \mu_{CB}, \xi \rangle \in CB_Z \), \( A \in \text{Hon} \) and \( \varphi \in \mathcal{L}_A^\dagger \) where spy does not occur. Then

1. \( \mu_{CB}, \xi \models \check{\varphi} \) if and only if \( \beta(\mu_{CB}, \xi) \models \check{\varphi} \);

2. \( \mu_{CB}, \xi \models \check{\alpha_A[\text{role}_A^\dagger(\sigma)]} \) if and only if \( \beta(\mu_{CB}, \xi) \models \check{\alpha_A[\text{role}_A^\dagger(\sigma)]} \).

**Proof:** The first condition follows from Lemma 2.2, given that for \( (\mu^\dagger, \xi^\dagger) = \beta(\mu_{CB}, \xi) \) we have by construction \( \mu_A = \mu_A^\dagger \) and \( \xi_A = \xi_A^\dagger \). The second condition follows from the first as roles never involve communication formulas. \( \square \)

In Lemma 5.7, we have that \( \check{\alpha_A[\varphi]}^\dagger \) equals \( \check{\alpha_A[\varphi]} \), and \( \check{\alpha_A[\text{role}_A^\dagger(\sigma)]}^\dagger \) equals \( \check{\alpha_A[\text{role}_A^\dagger(\sigma)]} \). Next, we prove a weaker version of this lemma for the intruder.

**Lemma 5.8** Let \( \langle \mu_{CB}, \xi \rangle \in CB_Z \). If \( \mu_{CB}, \xi \models \check{\alpha_Z[\text{knows}(M)]} \) then \( \beta(\mu_{CB}, \xi) \models \check{\alpha_Z[\text{knows}(M)]} \), for every message \( M \).

**Proof:** Let \( (\mu^\dagger, \xi^\dagger) = \beta(\mu_{CB}, \xi) \). If \( \mu_{CB}, \xi \models \check{\alpha_Z[\text{in}(A, M, B)]} \), then \( \mu^\dagger, \xi^\dagger \models \check{\alpha_Z[\text{P}_a \text{ in}(A, M, B)]} \), so, every message that \( Z \) could eventually spy on the CB model is received by him in the ZB model. Using axiom \( (K_Z) \), it follows that \( \mu_{CB}, \xi \models \check{\alpha_Z[\text{knows}(M)]} \) implies that \( \mu^\dagger, \xi^\dagger \models \check{\alpha_Z[\text{knows}(M)]} \), which concludes the proof. \( \square \)

Note that the converse of Lemma 5.8 fails since in \( \beta(\mu_{CB}, \xi) \) the intruder knows all messages sent from one agent to another, which is not the case for \( \langle \mu_{CB}, \xi \rangle \). From Lemmas 5.7 and 5.8, it follows that the transformation from CB models to ZB models is attack preserving, as formalized below.

**Proposition 5.9 (Attack preservation)** Let \( \langle \mu_{CB}, \xi \rangle \in CB_Z \). Then,

- an attack on \( \text{secre}_S(\sigma) \) at \( \langle \mu_{CB}, \xi \rangle \) implies an attack on \( \text{secre}_S(\sigma)^\dagger \) at \( \beta(\mu_{CB}, \xi) \);

- an attack on \( \text{ auth}_{A,B}^\dagger(\sigma) \) at \( \langle \mu_{CB}, \xi \rangle \) implies an attack on \( \text{ auth}_{A,B}^\dagger(\sigma)^\dagger \) at \( \beta(\mu_{CB}, \xi) \).
To prove the converse, we first define a model transformation \( \theta : ZB_\omega \rightarrow CB_\omega \). For each \( \langle \mu_{ZB}, \xi^i \rangle \in ZB_\omega \) with \( \mu_{ZB} = (\lambda^1, \alpha^1, \pi^1) \) we introduce explicit communication events local to the channel. Hence, let \( Fr = \{ e \in Ev^Z_B \mid \alpha^Z_B(e) = fresh(X), \) for some \( X \} \), \( Succ = \{ e \in Ev^Z_B \mid \alpha^Z_B(e) = in(A, M, B), \) for some \( A, M, B \} \), \( News = \{ s(e) \mid e \in Succ \} \), \( Pred = \{ e \in Ev^Z_B \mid \alpha^Z_B(e) = out(Z, M, A), \) for some \( M, A \} \), \( Newp = \{ p(e) \mid e \in Pred \} \), and \( Sync = Ev^Z_B \setminus Fr \). Note that \( s(e) \) and \( p(e) \) denote new events representing successor and predecessor events of \( e \) in the channel. Then, \( \theta(\mu_{ZB}, \xi^i) = (\mu, \xi) \) is as follows.

- \( \mu_A = \mu^1_A \) for every \( A \in Hon \);
- \( \mu_{CB} = (\lambda_{CB}, \alpha_{CB}, \pi_{CB}) \) with
  - \( \lambda_{CB} = \langle Ev_{CB}, \leq_{CB} \rangle \) where \( Ev_{CB} = (Ev^1_B \setminus Fr) \cup (News \cup Newp) \), and \( \rightarrow_{CB} \) is the successor relation obtained from \( \leq^*_Z \) by letting \( e \rightarrow_{CB} s(e) \) for every \( e \in Succ \), and \( p(e) \rightarrow_{CB} e \) for every \( e \in Pred \);
  - \( \alpha_{CB}(e) = \begin{cases} \alpha^1_Z(e) & \text{if } e \in Ev^1_Z \setminus Fr \\ leak & \text{if } e \in News \\ in(Z, M, A) & \text{if } e = p(e') \) and \( \alpha^1_Z(e') = out(Z, M, A) \)
- \( \mu_Z = (\lambda_Z, \alpha_Z, \pi_Z) \) where
  - \( \lambda_Z = \langle Ev_Z, \leq_Z \rangle \) is such that \( Ev_Z = Fr \cup News \cup Newp \), and \( \rightarrow_Z \) is obtained from \( \rightarrow^*_Z \) by replacing every \( e \in Succ \) by \( s(e) \), and every event in \( e \in Pred \) by \( p(e) \);
  - \( \alpha_Z(e) = \begin{cases} spy(M) & \text{if } e = s(e') \) and \( \alpha^1_Z(e') = in(A, M, B) \\ send(M, A) & \text{if } e = p(e') \) and \( \alpha^1_Z(e') = out(Z, M, A) \\ \alpha^1_Z(e) & \text{if } e \in Fr \)
- \( \pi_Z \) is inductively defined as expected, requiring \( \pi_Z(\emptyset) = \pi^1_Z(\emptyset) \).

- \( \xi_A = \xi^1_A \) for every \( A \in Hon \), \( \xi_{CB} = (\xi^1_Z \cap Ev_{CB}) \cup \{ s(e) \in News \mid e \in \xi^1_Z \} \cup \{ p(e) \in Newp \mid e \in \xi^1_Z \} \), and \( \xi_Z = (\xi^1_Z \cap Ev_Z) \cup \{ s(e) \in News \mid e \in \xi^1_Z \} \cup \{ p(e) \in Newp \mid e \in \xi^1_Z \} \).

Note that although \( \theta \) is not the inverse of \( \beta \), there is a clear relationship between \( \beta(\mu_{CB}, \xi) \) and \( (\mu, \xi) \). Namely, the former is precisely the \( CB \) interpretation structure in \( \beta^{-1}(\beta(\mu_{CB}, \xi)) \) that maximizes the activity of the intruder in terms of spying every possible message from the channel as soon as possible.

We claim that \( \theta(\mu_{ZB}, \xi^1) = (\mu, \xi) \) is a \( CB \) model. Recall the \( ZB \) model (the second) depicted in Figure 9 and consider, for instance, the event \( e \in Ev^1_B \cap Ev^1_Z \) such that \( \alpha^1_B(e) = send(M_2, A, A) \) and \( \alpha^1_Z(e) = in(B, M_2, A) \). Then, as also shown in the topmost \( CB \) model, we have two events \( e, s(e) \in Ev_{CB} \) with \( e \in Ev_B \) and \( s(e) \in Ev_Z \) such that \( \alpha_B(e) = send(M_2, A, A), \alpha_{CB}(e) = in(B, M_2, A), \alpha_{CB}(s(e)) = leak, \) and \( \alpha_Z(s(e)) = spy(M_2) \). Intruder sending actions are introduced similarly. In the translation of the global state, we add to \( \xi \) both the events corresponding to incoming messages from other principals that were already in \( \xi^1 \) and all outgoing messages from the intruder.

The proof of the following lemma is analogous to the proof of Lemma 5.7.

**Lemma 5.10** Let \( \langle \mu_{ZB}, \xi^1 \rangle \in ZB_\omega, A \in Hon, \) and \( \varphi \in L^Q_A \) where spy does not occur. Then

1. \( \theta(\mu_{ZB}, \xi^1) \models @_A[\varphi] \) if and only if \( \mu_{ZB}, \xi^1 \models @_A[\varphi]; \)
2. \( \theta(\mu_{ZB}, \xi^1) \models @_A[role^i_A(\sigma)] \) if and only if \( \mu_{ZB}, \xi^1 \models @_A[role^i_A(\sigma)]. \)

Observe that in the construction of the translated model, the intruder spies on every message that enters the channel. He therefore has the same knowledge as in the \( ZB \) model and we can establish the following result.

**Lemma 5.11** Let \( \langle \mu_{ZB}, \xi^1 \rangle \in ZB_\omega \). Then
Proof:

1. By the construction of $\alpha_Z$, the intruder in $\theta(\mu_{ZB}, \xi^\dagger)$ never receives messages. If $a$ is a fresh generation action, then the result is trivial. If $a = \text{send}(M, A)$, then it must be the label of an event $p(e')$ such that the label of $e'$ is $a^\dagger = \text{out}(Z, M, A)$.

2. The knowledge set computed with $(K_Z)$ at the given $\langle \mu_{ZB}, \xi^\dagger \rangle$ is the same that can be computed using $(K)$, for principal $Z$, by just exchanging each $\text{in}(A, M, B)$ with $\text{spy}(M)$, and recalling that $\pi_Z(\emptyset) = \pi_Z^1(\emptyset)$.

Lemmas 5.10 and 5.11 yield the following proposition, formalizing attack reflection.

**Proposition 5.12 (Attack reflection)**  Let $\langle \mu_{ZB}, \xi^\dagger \rangle \in ZB_Z$. Then,

- an attack on $\text{secr}_S(\sigma)^\dagger$ at $\langle \mu_{ZB}, \xi^\dagger \rangle$ implies an attack on $\text{secr}_S(\sigma)$ at $\theta(\mu_{ZB}, \xi^\dagger)$;

- an attack on $\text{auth}^{i,j}_{A,B}(\sigma)^\dagger$ at $\langle \mu_{ZB}, \xi^\dagger \rangle$ implies an attack on $\text{auth}^{i,j}_{A,B}(\sigma)$ at $\theta(\mu_{ZB}, \xi^\dagger)$.

As a corollary, we conclude that the two models are attack equivalent.

**Corollary 5.13** The models CB and ZB are attack equivalent.

5.2.2 Phase 2: Step compression

We now apply step compression, a model-reduction technique used to improve the efficiency of various protocol analysis tools, such as [3, 10, 23, 49, 52]. The essence of step compression is that when an honest principal receives a message, he immediately sends a reply. As shown in the third model of Figure 9, this reduction technique groups together entire protocol steps of honest agents into consecutive sequences of actions in the resulting model, as depicted inside the boxes.

Step-compressed models, or CZB models, are ZB models with an additional restriction. Recall that, for each $A \in \text{Hom}$, the sequence of actions labeling A’s local life-cycle must interleave prefixes of pairwise independent instantiations of protocol runs for principal $A$. Let $\text{run}_A^i(\sigma) = \text{step}_A^i,0(\sigma) \cdots \text{step}_A^i,m(\sigma) = (\text{act}_0, 1 \ldots \text{act}_{m,n}) \cdot (\text{act}_{1,1}, \ldots \text{act}_{1,n}) \cdots (\text{act}_{m,1}, \ldots \text{act}_{m,n})$ be such run and consider the prefix $(\ldots \text{act}_{k,0})$, with $k \leq m$ and $u_k \leq n_k$. Then, for each $0 \leq q \leq k$, we let $u_q = n_q$ and require that there are consecutive events $e_{i_q+1} \rightarrow A \cdots \rightarrow A e_{i_q+u_q}$ such that $\alpha_A(e_{i_q+1}) = \text{act}_{q,1}, \ldots, \alpha_A(e_{i_q+u_q}) = \text{act}_{q,1}$, i.e., the actions within a step cannot be interleaved with other actions. Below, we refer to the last event $e_{i_q+u_q}$ as the anchor event for step $q$ of the run. It should be clear that with the exception of the initial and final steps of a run, every non-empty intermediate step must start with a receiving action and end with a sending action. Thus, we will also require that the corresponding shared events on the intruder side are consecutive. That is, for each $1 \leq q \leq k$, such $q \neq m$ and $u_q = n_q \neq 0$, we require that $e_{i_q+1} \rightarrow Z e_{i_q+n_q}$.

At the specification level, this corresponds to adding axioms of the form:

$(\text{SC}_{A,q,u_0}^i(\sigma)) @A[\text{act}_{q,u_q} \Rightarrow Y(\text{act}_{q,u_q-1} \land Y(\ldots \land Y(\text{act}_{q,1} \ldots)))], \text{ for each } u_q \leq n_q \neq 0;

(\text{ZSC}_{A,q}^i(\sigma)) @Z[\circ_A[\text{act}_{q,u_q}] \Rightarrow Y \circ_A[\text{act}_{q,1}]], \text{ for each } 0 < q < m \text{ with } n_q \neq 0.$

We will show that if there is an attack on a security goal in an intruder-based model, then there is also a step-compressed model where the attack happens. The idea is to delay the events corresponding to each protocol step so that they are as close as possible to the step’s anchor event, and similarly for the intruder model, to delay the out actions so that they are next to the corresponding in actions, as described above. We now give the associated construction.
To establish the attack preservation results, we first define a model transformation $\beta : ZB \rightarrow CZB$. In this case, we do not consider states in the translation as the attacks on the different security goals may require constructing different global states. For each $\mu_{ZB} = (\lambda, \alpha, \pi) \in ZB$, we define the sets of floating events that must be reordered when constructing the step-compressed model as $Flt_A = \{e \in Ev_A \mid e \text{ is not an anchor event}\}$ for each $A \in Hon$, and $Flt_Z = Ev_Z \cap (\bigcup_{A \in Hon} Flt_A)$. Then, $\beta(\mu_{ZB}) = (\lambda', \alpha', \pi')$ is as follows:

- for $A \in Hon$, $\mu_A' = (\lambda_A', \alpha_A', \pi_A')$ is:
  - $\lambda_A' = (Ev_A', \leq_A')$ where $Ev_A' = Ev_A$ and $\rightarrow_A'$ results from restricting $\leq_A$ to anchor events in $Ev_A \setminus Flt_A$, and requiring that $e_{q,1} \rightarrow_A' e_{q,2} \rightarrow_A' \cdots \rightarrow_A' e_{q,u_a}$, where $e_{q,u_a}$ is the anchor event of the protocol step corresponding to the $\alpha_A$ labels of the events $e_{q,1} \leq_A e_{q,2} \leq_A \cdots \leq_A e_{q,u_a}$.
  - $\alpha_A' = \alpha_A$ and $\pi_A'(\emptyset) = \pi_A(\emptyset)$;

- for the intruder $Z$, $\mu_Z' = (\lambda_Z', \alpha_Z', \pi_Z')$ where:
  - $\lambda_Z' = (Ev_Z', \leq_Z')$ where $Ev_Z' = Ev_Z$ and $\rightarrow_Z'$ is the restriction of $\leq_Z$ to anchor events in $Ev_Z \setminus Flt_Z$, where we additionally require that $e_{q,1} \rightarrow_Z' e_{q,n}$ where $e_{q,1} \leq_Z e_{q,n}$ are the first and last events of a complete protocol step by some honest principal, and that $\rightarrow_Z'$ is compatible with the definition of $\leq_Z'$ for all the other floating events.
  - $\alpha_Z' = \alpha_Z$ and $\pi_Z'(\emptyset) = \pi_Z(\emptyset)$.

In $\beta(\mu_{ZB})$, protocol steps are grouped together as illustrated by the boxes in the CZB model in Figure 9. This grouping preserves the ordering of actions within runs and therefore also within steps. Only the interleaving of the different steps may change. Hence, $\beta(\mu_{ZB})$ is still a ZB model and, since it satisfies the restrictions, it is also a CZB model. To show that this transformation is attack preserving, we first prove two auxiliary results.

**Lemma 5.14** Let $\mu_{ZB} \in ZB$ and let $\xi$ be one of its global states. For each global state $\xi'$ of $\beta(\mu_{ZB})$ such that last($\xi_A$) = last($\xi_A'$), for $A \in Hon$, we have

$$\mu_{ZB}, \xi \models \Box_A [\text{role}_A(e)] \text{ if and only if } \beta(\mu_{ZB}), \xi' \models \Box_A [\text{role}_A'(e')] \text{.}$$

**Proof:** Assume that $\mu_{ZB}, \xi \models \Box_A [\text{role}_A(e)]$. Then, $\mu_{ZB}, \xi \models \Box_A \text{role}_A'(\sigma)$. Assume also that $\text{run}_A'(\sigma) = (act_1 \cdots act_n)$. Hence, there are $e_1, \ldots, e_n \in Ev_A$ such that $e_1 <_A \cdots <_A e_n$, $e_1 = \alpha_A(e_1) = act_1$, for $i \in \{1, \ldots, n\}$ and last($\xi_A$) = $e_n$. By construction, $e_1, \ldots, e_n \in Ev_A'$ and $\text{run}_A'(\sigma) = act_1$, for $i \in \{1, \ldots, n\}$. Furthermore, since $e_1, \ldots, e_n$ are associated with the same protocol run, then $e_1 <_A \cdots <_A e_n$ by the construction of $\rightarrow_A'$. Hence, as last($\xi_A'$) = last($\xi_A$) = $e_n$, it follows that $\beta(\mu_{ZB}), \xi' \models \Box_A [\text{role}_A'(\sigma)]$. The proof of the converse is similar. \hfill $\Box$

**Lemma 5.15** Let $A \in Hon$, $\mu_{ZB} \in ZB$, and $\xi$ be a global state of $\mu_{ZB}$ such that last($\xi_A$) = $e_A \in Ev_A \cap Ev_Z$ is an anchor event. Let $\xi'$ be the least global state of $\beta(\mu_{ZB})$ such that last($\xi'_A$) = $e_A$. Then $\xi'_A \subseteq \xi_A$ and for every $B \in Hon \setminus \{A\}$, if $e \in \xi_B$ is such that $e \in Ev_B \cap Ev_Z$ is an anchor event, then $e \in \xi_B$.

**Proof:** To start with, note that $\xi' = e_A \downarrow$ under the global ordering $\leq'$ . To show that $\xi'_A \subseteq \xi_A$, let $e \in \xi'_A$. As $e_A \in Ev_Z$, we have that $e \leq_Z e_A$. Assume, for the sake of contradiction, that $e \notin \xi_Z$. Then $e >_Z e_A$ and from the construction of $\rightarrow_Z'$ we conclude that $e$ is not an anchor event. Thus, in the construction, $e$ was moved towards its anchor event, which must also be after $e_A$. Hence $e >_Z^2 e_A$, which contradicts $e \leq_Z e_A$. Finally, let $B \in Hon \setminus \{A\}$ and $e \in \xi'_B$ be such that $e \in Ev_B \cap Ev_Z$ is an anchor event. As both $e, e_A \in Ev_Z$ are anchor events, then $e \leq_Z e_A$ and the definition of $\rightarrow_Z'$ imply that $e \leq_Z e_A$. Therefore, $e \in \xi_Z$ and also $e \in \xi_B$. \hfill $\Box$

We could also show, under the conditions of the previous lemma, that $\xi'_A \subseteq \xi_A$. However, this fact would not be useful for proving our envisaged result.
Lemma 5.16 (Attack preservation) Let \((\mu_{ZB}, \xi) \in ZB_{\Sigma}\). Then,

- an attack on \(secr{S}(\sigma)^\dagger\) at \((\mu_{ZB}, \xi)\) implies an attack on \(secr{S}(\sigma)\) at some state of \(\beta(\mu_{ZB})\);
- an attack on \(auth^{i,j}_{A,B}(\sigma)^\dagger\) at \((\mu_{ZB}, \xi)\) implies an attack on \(auth^{i,j}_{A,B}(\sigma)^\dagger\) at some state of \(\beta(\mu_{ZB})\).

Proof: Let \(\mu^p = \beta(\mu_{ZB})\). Assume that an attack on \(secr{S}(\sigma)^\dagger\) happens at the global state \(\xi\) of \(\mu_{ZB}\). Let \(\xi^p\) be the least global of \(\mu^p\) that contains \(\xi\). Then, \(\mu_{ZB}, \xi \vdash \bigwedge_{i=1}^j @_{A_i}[P_0 \text{ role}^j_{A_i}(\sigma)]\) and \(\mu_{ZB}, \xi \not\vdash \bigwedge_{B \in Princ} \big(\bigwedge_{A \in A_i} [P_0 \text{ role}^j_{A_i}(\sigma)]\big) \land M \in S \land @_B[\neg \text{knows}(M)]\). From Lemma 5.14, since \(\xi \subseteq \xi^p\), it follows that \(\mu^p, \xi^p \vdash \bigwedge_{i=1}^j @_{A_i}[P_0 \text{ role}^j_{A_i}(\sigma)]\). Furthermore, as \(\xi^p\) extends \(\xi\) and the labeling of events does not change, we also have that \(\mu^p, \xi^p \not\vdash \bigwedge_{B \in Princ} \big(\bigwedge_{A \in A_i} [P_0 \text{ role}^j_{A_i}(\sigma)]\big) \land M \in S \land @_B[\neg \text{knows}(M)]\).

Assume now that an attack on \(auth^{i,j}_{A,B}(\sigma)^\dagger\) happens at the global state \(\xi\) of \(\mu_{ZB}\). Let \(\xi^p\) be the least global state of \(\mu^p\) such that \(\text{last}(\xi^p) = \text{last}(\xi_A)\). Then, \(\mu_{ZB}, \xi \vdash @_A[\text{role}^j_{A_i}(\sigma)]\) and \(\mu_{ZB}, \xi \not\vdash @_B[P_0 \text{ send}(\sigma(M), A)]\). It follows from Lemma 5.14 that \(\mu^p, \xi^p \vdash @_A[\text{role}^j_{A_i}(\sigma)]\). Note also that any event of \(B \in \text{Hon} \setminus \{A\}\) labeled with a sending action is necessarily an anchor event shared with the channel. Thus, Lemma 5.15 guarantees that \(\mu^p, \xi^p \not\vdash @_B[P_0 \text{ send}(\sigma(M), A)]\). □

Hence, the two models are attack equivalent. Note that we do not state a reflection result because any CZB model is also a ZB model.

Corollary 5.17 The models ZB and CZB are attack equivalent.

5.2.3 Phase 3: Intruder-centric models

In our last translation step, we “forget” the honest agents and keep only the intruder, as illustrated in the last two models of Figure 9. An intruder-centric signature \(\Sigma_{ZC}\) is a distributed signature obtained from the network signature with just one agent identifier, \(Z\), for the intruder. Hence, \(\Sigma_{ZC} = \langle (Z), Act, Prop \rangle\) where, recalling the Alice-and-Bob-style protocol description above:

- \(Act_Z\) contains all the actions \(\text{trans}^j_{A,k}(\sigma)\), for each \(A \in \text{Princ}\), each protocol role \(0 \leq i \leq j\), each \(0 \leq k \leq m\), and each protocol instantiation \(\sigma\), as well as \(\text{fresh}(X)\) for \(X \in \text{Nonces} \cup \text{Sym}K \cup \text{Priv}K\);
- \(Prop_Z\) contains all the state propositions \(\text{knows}_{A}(M)\), for every \(A \in \text{Princ} \land M \in \text{Msg}\).

Note that we are considering the same network signature \((\text{Princ}, \text{Num})\) as before.

Before we proceed to establish the preservation results, we need some auxiliary notation:

- \(\text{recs}(\text{act}_1 \ldots \text{act}_k) = \{ M \mid \text{act}_i = \text{rec}(M) \text{ for some } 1 \leq i \leq k\}\);
- \(\text{snds}(\text{act}_1 \ldots \text{act}_k) = \{ M \mid \text{act}_i = \text{send}(M, A) \text{ for some } 1 \leq i \leq k \land A\}\);
- \(\text{fshs}(\text{act}_1 \ldots \text{act}_k) = \{ X \mid \text{act}_i = \text{fresh}(X) \text{ for some } 1 \leq i \leq k\}\).

We define the knowledge learned by a \(B \in \text{Hon}\) in a transaction as follows:

- \(\text{learn}_B(\text{trans}^j_{A,k}(\sigma)) = \begin{cases} \emptyset & \text{if } A \neq B \\ \text{recs}(\text{step}^j_{A,k}(\sigma)) \cup \text{fshs}(\text{step}^j_{A,k}(\sigma)) & \text{otherwise} \end{cases} \)
- \(\text{learn}_B(\text{fresh}(X)) = \emptyset\).

The knowledge learned by the intruder in each action, assuming \(\text{step}^j_{A,k}(\sigma) = \langle \text{act}_1 \ldots \text{act}_n \rangle\), is:

- \(\text{learn}_Z(\text{trans}^j_{A,k}(\sigma)) = \begin{cases} \emptyset & \text{if } \text{act}_n \neq \text{send}(M, A) \text{ for any } A, M \\ \text{snds}(\text{step}^j_{A,k}(\sigma)) & \text{otherwise} \end{cases} \)
• learn_Z(fresh(X)) = \{X\}.

Next, we rewrite all the relevant axioms in this new setting. These include axioms about knowledge, axioms to guarantee freshness and uniqueness of the data generated by each principal, and principal axioms among others.

(K*) \( \mu, \xi \models knows_A(M) \) iff \( M \in close(\{M' | \mu, \xi \models \forall Y knows_A(M')\}) \cup learn_A(\alpha(last(\xi))) \), for every state \( \xi \)

(F1^2_Z) \( @_Z [fresh(X) \Rightarrow Y(\neg knows_Z(M))] \)

(F2^2_Z) \( @_Z [fresh(X) \Rightarrow \bigwedge_{A \in Hon} \neg knows_A(M)] \)

(F1^1_A) \( @_Z [\text{trans}^1_{A,k}(\sigma) \Rightarrow Y(\neg knows_A(M^{'})]] \)

(F2^1_A) \( @_Z [\text{trans}^1_{A,k}(\sigma) \Rightarrow \bigwedge_{B \in Princ \setminus \{A\}} \neg knows_B(M^{'})] \)

Here \( M \) ranges over all messages such that \( cont(X) \cap cont(M) \neq \emptyset \) and \( M' \) ranges over all messages such that \( cont(X') \cap cont(M') \neq \emptyset \), for every \( X' \in \text{fshs}(\text{step}^1_{A,k}(\sigma)) \). Observe that these are exactly the same axioms as \((F1 - 2)\), but written in this new language.

Next, we rewrite the axioms about keys, where we assume that \( A, B \in Princ \) and \( A \neq B \).

(aKey^1) \( @_Z [s \models knows_A(K_A^{-1})] \)

(aKey^2) \( @_Z [s \models \neg knows_B(M)], \) for every \( M \) containing \( K_A^{-1} \)

(sKey^1.1) \( @_Z [s \models knows_A(K_{AB})] \)

(sKey^1.2) \( @_Z [s \models knows_B(K_{AB})] \)

(sKey^2) \( @_Z [s \models \neg knows_C(M)], \) for every \( C \in Princ \setminus \{A, B\} \) and every \( M \) containing \( K_{AB} \)

We may also assume, for simplicity, that all principals \( A, B \in Princ \) know each other’s names and public keys from the start.

(N) \( @_Z [s \models knows_A(B)] \)

(PK*) \( @_Z [s \models knows_A(K_B)] \)

In order for \( \mu \) to be a model of the protocol, we require that it satisfies the above axioms and also that the projection of the trace on each honest agent \( A \), after expanding the actions, is still a legal run, i.e. an interleaving of prefixes of runs (as before). We delay the details of this expansion until the definition of the model transformation \( \theta \) from ZC models to CZB models.

Observe that our notion of model requires, for every \( \text{trans}^1_{A,q}(\sigma) \) and \( \text{trans}^1_{A,k}(\sigma) \) with \( k < q \), that \( @_Z[\text{trans}^1_{A,q}(\sigma) \Rightarrow P \text{trans}^1_{A,k}(\sigma)] \).

The change in the language also affects how our security goals are written. Consider first the notion of role. In this setting, if \( run^i(\sigma) = step^i_{A,k_1} \cdots step^i_{A,k_q} \) then

\[ role^i_A(\sigma) \equiv \text{trans}^i_{A,k_q}(\sigma) \land P(\text{trans}^i_{A,k_{q-1}}(\sigma) \land P(\ldots \land P \text{trans}^i_{A,k_1}(\sigma))) \]

To express that the messages in a finite set \( S \) will remain a shared secret between participants \( A_1, \ldots, A_j \) after the complete execution of a protocol under the instantiation \( \sigma \), with \( \sigma(a_i) = A_i \), we write:

\[ @_Z \left[ \bigwedge_{i=1}^{j} P_c \ role^i_{A_i}(\sigma) \Rightarrow \bigwedge_{M \in S, B \in Princ \setminus \{A_1, \ldots, A_j\}} \neg knows_B(M) \right] \]

and denote this formula by \( \text{secr}_S(\sigma) \).
As for authentication properties, let $\sigma$ be a protocol instantiation such that $\sigma(a_i) = A \in \text{Hon}$ and $\sigma(a_j) = B \in \text{Hon}$. Then the property that $A$ authenticates $B$ in role $j$ at step $q$ of the protocol can be defined by the following formula $\text{auth}_{A,B}^j(\sigma)^*$, assuming that $\text{msg}_q$ of the protocol requires that $B$ sends the message $M$ to $A$:

$$\forall Z [\text{role}_A^j(\sigma)^* \Rightarrow P_\sigma \text{trans}_{B,q}^j(\sigma)]$$

Note that it must be the case that $\text{step}_{B,q}^j(\sigma) = \langle \text{act}_1, \ldots, \text{act}_k \rangle$ and $\text{act}_k = \text{send}(M, A)$.

We now proceed to prove that the two models are attack-equivalent. We start by defining the model transformation $\theta : ZC \rightarrow CZB$ from $ZC$ models to $SC$ models, which is considerably easier than the converse direction. The following notation will be useful. Let $\mu_{ZC} = (\lambda^*, \alpha^*, \pi^*) \in ZC$ with $\lambda = \langle Ev^*, \Rightarrow \rangle$ and $A \in \text{Hon}$. We denote by $Ev_A$ the events of $Ev^*$ that involve actions from $A$ and we will need as many as the number of atomic actions in each of the corresponding steps.

Hence, we define

$$Ev_A = \{(e, k) \mid e \in Ev^*_Z, \alpha^*(e) = \text{trans}_{A,q}^j(\sigma) \text{ and } 1 \leq k \leq |\text{step}_{A,q}^j(\sigma)|\}.$$  

We also need a similar construction for the intruder:

$$Ev_Z = \{(e, 1) \mid e \in Ev^*_Z, \alpha_Z^*(e) = \text{trans}_{A,q}^j(\sigma) \text{ and } (\text{step}_{A,q}^j(\sigma))_1 = \text{rec}(M) \text{ for some } M \} \cup \{(e, k) \mid e \in Ev^*_Z, \alpha_Z^*(e) = \text{trans}_{A,q}^j(\sigma), |\text{step}_{A,q}^j(\sigma)| = k \text{ and } \text{step}_{A,q}^j(\sigma)_k = \text{send}(M, B) \text{ for some } B, M \} \cup \{(e, 1) \mid e \in Ev^*_Z \text{ and } \alpha_Z^*(e) = \text{fresh}(X) \text{ for some } X\}.$$ 

Then $\theta(\mu_{ZC}) = (\lambda, \alpha, \pi)$ is as follows:

- $\mu_A = (\lambda_A, \alpha_A, \pi_A)$, for each $A \in \text{Hon}$, is
  - $\lambda_A = \langle Ev_A, \leq_A, \text{rec}_A \rangle$ such that $(e_1, k_1) \leq_A (e_2, k_2)$ if $e_1 \leq^*_Z e_2$, or $e_1 = e_2$ and $k_1 < k_2$;
  - $\alpha_A((e, k)) = (\text{step}_{A,q}^j(\sigma))_k$ if $\alpha^*(e) = \text{trans}_{A,q}^j(\sigma)$;
  - $\pi_A(\emptyset) = \{\text{knows}(M) \mid \text{knows}_A(M) \in \pi_Z(\emptyset)\}$;

- $\mu_Z = (\lambda_Z, \alpha_Z, \pi_Z)$ is
  - $\lambda_Z = \langle Ev_Z, \leq_Z, \text{rec}_Z \rangle$ such that $(e_1, k_1) \leq_Z (e_2, k_2)$ if $e_1 \leq^*_Z e_2$, or $e_1 = e_2$ and $k_1 < k_2$;
  - $\alpha_Z((e, k)) = \begin{cases} \text{out}(Z, M, A) & \text{if } \alpha_A((e, k)) = \text{rec}(M), \text{ for the unique } A \in \text{Hon} \text{ such that } (e, k) \in Ev_A \\ \text{in}(A, M, B) & \text{if } \alpha_A((e, k)) = \text{send}(M, B), \text{ for the unique } A \in \text{Hon} \text{ such that } (e, k) \in Ev_A \\ \alpha_Z^*(e) & \text{otherwise}; \end{cases}$
  - $\pi_Z(\emptyset) = \{\text{knows}(M) \mid \text{known}_Z(M) \in \pi_Z(\emptyset)\}$.

Our construction of $CZB$ models is quite intuitive. We simply expand each action $\text{trans}_{A,q}^j(\sigma)$ into its corresponding atomic actions. For instance, in Figure 9, $\text{trans}_{A,0}^{\text{init}}$ in the $ZC$ model is expanded to $\text{fresh}(F_1) \Rightarrow \text{send}(M_1, B)$ for $A$ and to $\text{in}(A, M_1, B)$ for $Z$ in the corresponding $CZB$ model.

Given the above translation, it is straightforward to define a translation for states. Given a $ZC$ state $\xi^*$ then $\xi$ is the $CZB$ state such that $\xi_A = \{(e, k) \in Ev_A \mid e \in \xi^*_Z\}$ for each $A \in \text{Princ}$.

Next, we prove the reflection of attacks on security goals from $ZC$ models to $CZB$ models. We start with some preliminary results.

**Lemma 5.18** Let $\mu_{ZC} \in ZC$ and let $\xi^*$ be one of its states. Then, for every $A \in \text{Princ}$

$$\mu_{ZC}, \xi^* \vdash @Z [\text{knows}_A(M)] \text{ if and only if } \theta(\mu_{ZC}), \xi \vdash @A [\text{knows}(M)], \text{ for every } M.$$
Proof: Assume that \( A \in \text{Hon} \) and let \( \mu = \theta(\mu_{ZC}) \). Then, \( \mu_{ZC}, \xi^* \models @Z[\text{knows}_A(M)] \) if and only if \( \text{knows}_A(M) \in \pi^*_Z(\xi^*) \) if \( M \in \text{close}(M' \mid \text{knows}_A(M') \in \pi^*_Z(\emptyset)) \cup \bigcup_{e \in \mathcal{E}_A} \text{learn}_A(\alpha^*_Z(e)) \) if \( M \in \text{close}(M' \mid \text{knows}(M') \in \pi^*_Z(\emptyset)) \cup \bigcup_{e \in \mathcal{E}_A} \text{recs}(\alpha_A(e)) \cup \text{fsks}(\alpha_A(e)) \) if \( \text{knows}(M) \in \pi_A(\xi_A) \) if \( \mu, \xi \models @A[\text{knows}(M)] \).

For the intruder, we have that \( \mu_{ZC}, \xi^* \models @Z[\text{knows}_Z(M)] \) if \( \text{knows}_Z(M) \in \pi^*_Z(\xi^*) \) if \( M \in \text{close}(M' \mid \text{knows}_Z(M') \in \pi^*_Z(\emptyset)) \cup \bigcup_{e \in \mathcal{E}_A} \text{learn}_A(\alpha^*_Z(e)) \) if \( M \in \text{close}(M' \mid \text{knows}(M') \in \pi^*_Z(\emptyset)) \cup \bigcup_{e \in \mathcal{E}_A} \text{recs}(\alpha_A(e)) \cup \text{fsks}(\alpha_A(e)) \) if \( \text{knows}(M) \in \pi_A(\xi_A) \) if \( \mu, \xi \models @Z[\text{knows}(M)] \).

Lemma 5.19 Let \( \mu_{ZC} \in ZC \) and let \( \xi^* \) be one of its states. Then, for every \( A \in \text{Hon} \)

\[
\mu_{ZC}, \xi^* \models @Z[\text{trans}^{i_A}_{A,q}(\sigma)] \text{ if and only if } \theta(\mu_{ZC}), \xi \models \text{SC}^{i_A}_{A,q,u_A}(\sigma).
\]

Proof: Let \( \mu = \theta(\mu_{ZC}) \). Then, \( \mu_{ZC}, \xi^* \models @Z[\text{trans}^{i_A}_{A,q}(\sigma)] \) if \( \text{trans}^{i_A}_{A,q}(\sigma) \) iff, by definition of \( \mu_A \), there exist events \((e,1), \ldots, (e,u_A) \in \text{Ev}_A \) such that \((e,1) \rightarrow_A \ldots \rightarrow_A (e,u_A) \) and \( \alpha_A((e,j)) = \text{act}_{q,j}, \) for \( j \in \{1, \ldots, u_A\} \), if \( \mu, \xi \models \text{SC}^{i_A}_{A,q,u_A}(\sigma) \), observing that \( \text{last}(\xi_A) = (e,u_A) \).

An immediate consequence of the previous result is the following lemma.

Lemma 5.20 Let \( \mu_{ZC} \in ZC \) and let \( \xi^* \) be one of its states. Then, for every \( A \in \text{Hon} \)

\[
\mu_{ZC}, \xi^* \models @Z[\text{role}^i_{A,B}(\sigma)^*] \text{ if and only if } \theta(\mu_{ZC}), \xi \models @A[\text{role}^i_A(\sigma)].
\]

These results allow us to establish the following proposition.

Proposition 5.21 (Attack reflection) Let \( \mu_{ZC} \) and let \( \xi^* \) be one of its local states. Then,

- an attack on \( \text{secrets}(\sigma)^* \) at \( (\mu_{ZC}, \xi^*) \) implies an attack on \( \text{secrets}(\sigma)^{i_A} \) at \( (\theta(\mu_{ZC}), \xi) \);
- an attack on \( \text{auth}^{i,j,q}_{A,B}(\sigma)^* \) at \( (\mu_{ZC}, \xi^*) \) implies an attack on \( \text{auth}^{i,j,q}_{A,B}(\sigma)^{i_A} \) at \( (\theta(\mu_{ZC}), \xi) \).

Proof: Let \( \mu = \theta(\mu_{ZC}) \). Assume that an attack on \( \text{secrets}(\sigma)^* \) happens at the global state \( \xi^* \) of \( \mu_{ZC} \). Then, it must be the case that \( \mu_{ZC}, \xi^* \models @Z[\bigwedge_{A \in A} \text{role}^i_A(\sigma)^{i_A}] \) and also that \( \mu_{ZC}, \xi^* \models @Z[\bigwedge_{M \in \text{E}_S} \bigwedge_{B \in \text{BPriv}_{\{A_1, \ldots, A_n\}}} \neg \text{knows}_B(M)] \). From Lemma 5.20, it follows that \( \mu, \xi \models \bigwedge_{A \in A} @A[\text{role}^i_A(\sigma)] \), and from Lemma 5.18, it follows that \( \mu, \xi \models \bigwedge_{M \in \text{E}_S} \bigwedge_{B \in \text{BPriv}_{\{A_1, \ldots, A_n\}}} \neg @B[\text{knows}(M)] \). Hence, an attack on \( \text{secrets}(\sigma)^{i_A} \) happens at the global state \( \xi \) of \( \mu \).

Assume now that an attack on \( \text{auth}^{i,j,q}_{A,B}(\sigma)^* \) happens at the global state \( \xi^* \) of \( \mu_{ZC} \), for \( B \in \text{Hon} \). Then, \( \mu_{ZC}, \xi^* \models @Z[\text{role}^i_A(\sigma)^*] \) and \( \mu_{ZC}, \xi^* \models @Z[\text{trans}^{i_A}_{B,q}(\sigma)] \). From Lemma 5.20, it follows that \( \mu, \xi \models @A[\text{role}^i_A(\sigma)] \), and from Lemma 5.19, it follows that \( \mu, \xi \models @B[\text{SC}^{i_A}_{B,q,u_B}(\sigma)] \). Note that, by the construction of \( \mu \) and \( \xi \), and by the fact that \( B \in \text{Hon} \) and, thus, is following his role in the protocol, it cannot be the case that \( \mu, \xi \models @B[\text{act}_{q,u_B}] \) and \( \mu, \xi \models @B[\text{act}_{q,u_B} \Rightarrow (\text{act}_{q,u_B-1} \wedge \text{Y} \cdots \wedge \text{Y} \text{act}_{q,1})] \). Furthermore, by the definition of \( \text{auth}^{i,j,q}_{A,B}(\sigma)^* \), we have that \( \text{act}_{q,u_B} = \text{send}(M,A) \). Hence, \( \mu, \xi \models @B[\text{act}_{q,u_B}] \). Thus, an attack on \( \text{auth}^{i,j,q}_{A,B}(\sigma)^* \) happens at the global state \( \xi \) of \( \mu \).

Now, we prove the converse: if an attack on a security goal occurs in a CZB model it is possible to mimic that attack in a ZC model. In this case, we consider two different model transformations depending on the type of attack that we are considering. This has to do with the way we treat incomplete steps. In first transformation, we include the incomplete steps in the resulting model as we will use this translation to show attack preservation on secrecy properties. Then, we consider a second transformation where we forget all the incomplete steps and use this to show attack preservation on authentication properties.

Let \( \mu_{CZB} = (\lambda, \alpha, \sigma) \in CCB \). We consider the following sets of events:

5Here, we write \( \text{SC}^{i_A}_{B,q,u_B}(\sigma) \) for \( @B[\text{act}_{q,u_B} \Rightarrow (\text{act}_{q,u_B-1} \wedge \text{Y} \cdots \wedge \text{Y} \text{act}_{q,1})] \).
• $E_{v_f} = \{ e \in E_v \mid \alpha_z(e) = \text{fresh}(X), \text{ for some } X \}$,
• $E_{v_e} = \bigcup_{A \in \text{Hon}} \{ e \in E_v \mid e \text{ is an anchor event in some run of } A \}$,
• $E_{v_a} = \bigcup_{A \in \text{Hon}} \{ e \in E_v \mid e \text{ is an anchor event in some run of } A \}$.

Note that the set $E_{v_e}$ contains the (anchor) events corresponding to all the steps that were completed. In contrast, $E_{v_a}$ contains all (anchor) events of steps that were started, including the ones corresponding to incomplete steps. Observe that if a step was completed, then the anchor event is shared between the agent and the intruder (the last action of a step is always a send or a rec). If the step was not completed then the anchor event might not be shared with the intruder, but will always be an event of the agent performing that step. In fact, $E_{v_a} \subseteq E_{v_e}$.

The first model transformation that we consider is $\beta^1 : CZB \rightarrow ZC$, which we use to show the preservation of attacks on secrecy properties. In this case, $\beta^1(\mu_{CZB}) = (\lambda^*, \alpha^*, \pi^*)$ is as follows:

• $E_{v_f} = E_{v_e} \cup E_{v_a}$ and $\leq_z$ is any linearization of $E_{v_z}$ compatible with $\leq_z$;
• $\alpha_z(e) = \begin{cases} \alpha_z(e) & \text{if } e \in E_{v_f} \\ \text{trans}_{A,q}(\sigma) & \text{if } e \in E_{v_a} \text{ and } e \text{ occurs in } \text{step}_{A,q}(\sigma) \end{cases}$;
• $\pi_z(\emptyset) = \bigcup_{A \in \text{Princ}} \{ \text{knows}(M) \mid \text{knows}(M) \in \pi_A(\emptyset) \}$.

In the following, given a global state $\xi$ of $\mu_{CZB}$, we denote by $\xi^*$ the global state of $\beta^1(\mu_{CZB})$ such that $\xi^* = (\xi \cap E_{v_f}) \cup (\bigcup_{A \in \text{Hon}} (\xi_A \cap E_{v_a}^*))$.

**Lemma 5.22** Let $\mu_{CZB} \in CZB$ and let $\xi$ be one of its local states. Then, for every $A \in \text{Hon}$,

$$\text{if } \mu_{CZB}, \xi \models \mathbf{SC}^i_{A,q,u}(\sigma) \text{ then } \beta^1(\mu_{CZB}), \xi^* \models \mathbf{P}_{o} \text{trans}_{A,q}(\sigma) \text{.}$$

**Proof:** Let $\mu^* = \beta^1(\mu_{CZB})$. If $\mu_{CZB}, \xi \models \mathbf{SC}^i_{A,q,u}(\sigma)$, then there exist $e_1 \rightarrow_A \cdots \rightarrow_A e_{u,q}$ such that $\alpha_A(e_j) = (\text{step}_{A,q}(\sigma))$). In particular, $e_{u,q}$ is an anchor and thus $\xi_{e_{u,q}} \in \xi^*$. Furthermore, $\alpha^*_Z(e_{u,q}) = \text{trans}_{A,q}(\sigma)$ and so, $\mu^*, \xi^* \models \mathbf{P}_{o} \text{trans}_{A,q}(\sigma) \text{.}$

**Lemma 5.23** Let $\mu_{CZB} \in CZB$ and let $\xi$ be one of its local states. Then, for every $A \in \text{Hon}$,

$$\text{if } \mu_{CZB}, \xi \models \mathbf{A}[\text{role}_{A}(\sigma)] \text{ then } \beta^1(\mu_{CZB}), \xi^* \models \mathbf{Z}[\mathbf{P}_{o} \text{role}_{A}^i(\sigma)] \text{.}$$

**Proof:** Let $\beta^1(\mu_{CZB}) = \mu^*$ and $\text{run}_{A}^i(\sigma) = \text{step}_{A,0}(\sigma) \cdot \cdots \cdot \text{step}_{A,m}(\sigma) = (\text{act}_{0,1} \cdot \cdots \cdot \text{act}_{0,n_0}, \text{act}_{1} \cdot \cdots \cdot \text{act}_{1,n_1}, \cdots, \text{act}_{m-1} \cdot \cdots \cdot \text{act}_{m,m_m})$. If $\mu_{CZB}, \xi \models \mathbf{A}[\text{role}_{A}(\sigma)]$ then there are $e_{0,1} \rightarrow_A \cdots \rightarrow_A e_{0,n_0} \leq A \cdots \leq A e_{m,1} \rightarrow_A \cdots \rightarrow_A e_{m,m_n}$ such that $\alpha_A(e_{j,k}) = \text{act}_{j,k}$. In particular, $e_{j,n_j}$ with $j \in \{0, \ldots , m\}$, are all anchor events and are all shared with $Z$. Hence, $e_{0,n_0} \leq Z \cdots \leq Z e_{m,m_n}$ and, furthermore, $e_{j,n_j} \in E_{v_z}$, for $j \in \{0, \ldots , m\}$. Thus, $e_{j,n_j} \in \xi_{e}^*, \alpha_z(e_{j,n_j}) = \text{trans}_{A,j}(\sigma)$ and $e_{0,n_0} \leq Z \cdots \leq Z e_{m,m_n}$. Hence $\mu^*, \xi^* \models \mathbf{Z}[\mathbf{P}_{o} \text{role}_{A}^i(\sigma)] \text{.}$

**Lemma 5.24** Let $\mu_{CZB} \in CZB$ and let $\xi$ be one of its local states. Then, for every $A \in \text{Princ}$,

$$\text{if } \mu_{CZB}, \xi \models \mathbf{A}[\text{knows}(M)] \text{ then } \beta^1(\mu_{CZB}), \xi^* \models \mathbf{Z}[\mathbf{A}[\text{knows}(M)]] \text{.}$$

**Proof:** We start by proving that $\bigcup_{e \in \xi_A} \text{recs}(\alpha_A(e)) \cup \text{fshs}(\alpha_A(e)) \subseteq \bigcup_{e \in \xi_{e}^*} \text{learn}_{A}(\alpha^*_Z(e))$. Let $e \in \xi_A$ such that $\alpha_A(e)$ occurs in some $\text{step}_{A,q}(\sigma)$. If $e$ is an anchor event, then $e \in \xi_{e}^*$ and $\alpha^*_Z(e) = \text{trans}_{A,q}(\sigma)$. Hence, $\text{recs}(\alpha_A(e)) \cup \text{fshs}(\alpha_A(e)) \subseteq \text{learn}_{A}(\alpha^*_Z(e))$. If $e$ is not an anchor event, let $e'$ be its anchor event. Again, $e' \in \xi_{e}^*$ and $\alpha^*_Z(e') = \text{trans}_{A,q}(\sigma)$, and so, $\text{recs}(\alpha_A(e)) \cup \text{fshs}(\alpha_A(e)) \subseteq \text{learn}_{A}(\alpha^*_Z(e'))$. The rest of the proof follows as in Lemma 5.18 using this result. \(\Box\)
Proposition 5.25 (Attack preservation – secrecy) Let $μ_{CZB} \in CZB$ and let $ξ$ be one of its local states. Then an attack on $secr_s(σ)^1$ at $(μ_{CZB}, ξ)$ implies an attack on $secr_s(σ)^*$ at $(β^1(μ_{CZB}), ξ^*)$.

Proof: The proof of this result is similar to the proof of Proposition 5.21 using Lemma 5.23 and Lemma 5.24.

We now focus on authentication properties, where the model translation will only consider the anchor events from completed steps. All incomplete steps will be ignored. In this case, $β^2(μ_{CZB}) = \langle μ^*, α^*, π^* \rangle$ is as follows:

- $Ev^*_Z = Ev^*_A \cup Ev^*_C$ and $≤ Z^*$ is the restriction of $≤ Z$ to $Ev^*_Z$;
- $α^*_Z$ and $π^*_Z$ are as in $β^1$.

In the following, given a global state $ξ$ of $μ_{CZB}$ and $A ∈ Hon$, we denote by $ξ^*_A$ the global state of $β^2(μ_{CZB})$ such that $(ξ^*_A)_Z = last(ξ_A) ↓ (in \ β^2(μ_{CZB}))$.

Lemma 5.26 Let $A ∈ Hon$, $μ_{CZB} ∈ CZB$, and $ξ$ be a global state of $μ_{CZB}$ such that $last(ξ_A) ∈ Ev_A ∩ Ev_Z$ is an anchor event. Then, $(ξ^*_A)_Z ≤ Z$, and for every $B ∈ Hon \ \{A\}$, if $e ∈ (ξ^*_A)_Z$ is such that $e ∈ Ev_B ∩ Ev_Z$ is an anchor event, then $e ∈ ξ_B$.

Proof: The condition on $Z$ is straightforward by construction of $Ev^*_Z$. Let $e_A = last(ξ_A)$. As both $e$ and $e_A$ are anchor events then $e, a_A ∈ Ev^*_Z$. If $e ∈ (ξ^*_A)_Z$, then it must be the case that $e ≤ Z e_A$. Hence, $e ≤ Z e_A$ which implies that $e ∈ ξ_Z$ and, so, $e ∈ ξ_B$.

Lemma 5.27 Let $μ_{CZB} ∈ CZB$ and let $ξ$ be one of its local states. Then, for every $A ∈ Hon$ and $j ∈ \{1, \ldots, |step^i_A(σ)|\}$,

- $β^2(μ_{CZB}), ξ^*_A \models @Z[\mathsf{P}_o trans^i_A(σ)]$ then $μ_{CZB}, ξ \models @A[\mathsf{P}_o(step^i_A(σ))_j]$.

Proof: Let $μ^* = β^2(μ_{CZB})$ and assume that $step^i_A(σ) = \langle act_1 \ldots act_k \rangle$. If $μ^*, ξ^*_A \models @Z[\mathsf{P}_o trans^i_A(σ)]$ then there is an $e ∈ (ξ^*_A)_Z$ such that $α^*_Z(e) = trans^i_A(σ)$. Furthermore, $e ∈ Ev_A ∩ Ev_Z$ and $e$ is an anchor event, and so, $e ∈ ξ_A$. By the definition of anchor events, there are $e_1 A \rightarrow \cdots \rightarrow A e_k = e$ such that $α_A(e_j) = act_j$. Hence, $e_j ∈ ξ_A$ and the result follows.

Lemma 5.28 Let $μ_{CZB} ∈ CZB$ and let $ξ$ be one of its local states. Then, for every $A ∈ Hon$,

- $β^2(μ_{CZB}), ξ^*_A \models @Z[\mathsf{role}_A^i(σ)^*]$ if $μ_{CZB}, ξ \models @A[\mathsf{role}_A^i(σ)]$.

Proof: Let $β^2(μ_{CZB}) = μ^*$ and $run^i_A(σ) = step^i_A,0(σ) \cdots step^i_A,|act_0(σ)| (σ) = \langle act_{0,1} \cdots act_{0,|act_0(σ)|}, act_{1,1} \cdots act_{1,|act_1(σ)|}, \ldots, act_{m,n,1} \cdots act_{m,n,|act_m,n(σ)|} \rangle$. Assume that $μ_{CZB}, ξ \models @A[\mathsf{role}_A^i(σ)]$. Then, there are $e_{0,1} A \rightarrow A e_{0,n_0} ≤ A \cdots ≤ A e_{m,1} A \rightarrow A e_{m,n,m}$ such that $α_A(e_j) = act_j$ and such that $e_j,n ∈ Ev_A ∩ Ev_Z$ are anchor events. Then, by construction, $e_{0,n_0} ≤ Z \cdots ≤ Z e_{m,n,m}$ and $α^*_Z(e_j,n) = trans^i_A(σ)$. This implies that $μ^*, ξ^*_A \models @Z[\mathsf{role}_A^i(σ)^*]$, since $last((ξ^*_A)_Z) = e_{m,n,m}$.

The proof of the converse is similar.

Proposition 5.29 (Attack preservation – authentication) Let $μ_{CZB} ∈ CZB$ and let $ξ$ be one of its local states. Then an attack on $auth^i_{A,B}(σ)^1$ at $(μ_{CZB}, ξ)$ implies an attack on $auth^i_{A,B}(σ)^*$ at $(β^2(μ_{CZB}), ξ)$.

Proof: The proof of this result is similar to the proof of Proposition 5.21 using Lemma 5.28 and Lemma 5.27.

From Proposition 5.21, Proposition 5.25, and Proposition 5.29, we have the following corollary.

Corollary 5.30 The models $CZB$ and $ZC$ are attack equivalent.
6 Related work and conclusions

Communication, distribution, and cryptography are the essential ingredients of security protocols. Many attacks on security protocols arise from problems in communication and distribution, rather than cryptography itself. Hence, we follow the approach often taken in the formal methods community of abstracting away cryptographic details by assuming perfect black-box cryptography. The remaining ingredients — communication and distribution — are precisely the central concepts underlying DTL, which suggests DTL’s suitability for this domain. DTL is neutral with respect to the kinds of interpretation structures it formalizes and, as we showed through our case studies, by choosing different signatures and axioms, we can define theories that are well suited for formalizing and reasoning about different application domains and problems in security protocol analysis.

DTL is closely related to the family of temporal logics whose semantics are based on the models of true concurrency, introduced and developed in [40, 41, 57]. DTL was proposed in [34] as a logic for specifying and reasoning about distributed information systems and several versions were given, reflecting different perspectives on how non-local information can be accessed by each agent [8, 16, 17]. In this paper, we use the simplest and most expressive formulation, from [8]. We stick with a propositional language, which suits our purposes, as we describe our schema axioms by taking advantage of explicit meta-level quantifications and of the inductive definition of closed sets of messages.

Of course, there are other formalisms for modeling distributed, communicating systems. A key difference is that DTL provides not just a modeling language (as process algebras also do, for example) but also a logic for reasoning about systems. Reasoning about local temporal properties of distributed agents could also be performed in a linear temporal logic over linearizations of the distributed models. However, this would come at the price of readability and simplicity. In contrast, DTL is simple and robust in the sense that formulas are invariant with respect to different linearizations. We have taken advantage of this in the proofs given in this paper.

With respect to formalizing and reasoning about security protocols and associated models, a large number of logics, formalisms, and tools have been proposed in recent years, e.g. [3, 5, 10, 12, 22, 30, 62, 63, 38, 42, 46, 48, 52, 55, 59, 60, 64, 65]. We will not compare with those logics that have been proposed solely as an object logic for protocol verification, e.g. [13, 15, 21, 38, 56, 58]. Rather, we compare with formalisms and logics that can be used to establish metatheoretic properties of protocol models or even relate protocol models.

The semantics of DTL is based on event structures. The most closely related formalism from the security community is strand spaces [36, 60, 63], which is widely used to analyze properties and models of security protocols. In [17], we formally investigated the relationship between the interpretation structures of our DTL network models and strand spaces. This comparison yields, transitively, a comparison with the other approaches that have been related to strand spaces, e.g. [20, 28, 29, 37]. Our results show that DTL network models and strand-space models are compatible, although they offer different views of protocol executions. We defined property-preserving, back-and-forth translations between models in our logic and strand-space models. This is nontrivial as, despite the similarities between the two formalisms (for example, both are based on partially-ordered sets of events with labeling information), there are substantial differences. These differences concern the way the principals and the intruder executing a protocol are represented, the way communication is formalized, and the locality of information. While carrying out semantic reasoning directly in terms of our interpretation structures is not that different than reasoning about bundles in the strand space approach, a fundamental difference is that strand spaces do not provide a logic. Hence, strand spaces lack a means for specifying classes of models axiomatically, a deductive system, a property specification language, and the ability to relate models based on the properties they preserve, possibly under formula translations.

The idea of using DTL to investigate general metatheoretic properties of security protocol models and model simplification techniques was first explored in [16]. In this preliminary work, we used DTL to formalize and establish the correctness of two model-simplification techniques. We first proved that one intruder is enough, namely that it is sufficient to consider one Dolev-Yao intruder instead of multiple intruders. Second, we proved the correctness of a predatory intruder,
which is an intruder with restricted behavior, e.g. who only sends messages that are immediately received and processed by honest agents. Lemma 4.1 from the current paper generalizes not only a similar result from [16] but also the protocol-independent secrecy results of [27, 47], which capitalize on the notion of honest ideals on strand spaces introduced in [62].

The step-compression technique that we considered in Section 5.2 is used in several other approaches, such as [3, 10, 52]. We have begun applying our logic to validate other techniques that can be used to improve the performance of analysis tools for security protocols, such as the partial-order techniques developed for the OFMC model checker [9, 53].

Numerous formal models have been proposed for reasoning about communication channels in security protocols and services, e.g. [1, 4, 14, 31, 32, 45, 51]. For example, [51] gives two different abstract models for channels that are authentic, confidential, or secure: one that represents the ideal functionality of the channel and a second that employs concrete cryptographic messages to realize the channel properties. These two models are then shown equivalent under suitable assumptions. Our metareasoning results are in this spirit, in particular the proof of equivalence of the two models for guaranteeing message-origin authentication. It will be interesting to see how far we can take our approach in this regard. That is, whether and how our approach can be applied not only to obtain other such simulation-based results, but also to reason about protocol compositionality as done, for instance, in [6, 11, 19] at the cryptographic level and in [51] as well as in [1, 2, 14, 25, 31, 32, 35, 56] at a symbolic, black-box cryptography level. The combination of results, techniques, and tools from these and related approaches will play an important role in consolidating research in security protocol analysis.

We close with a final word on our use of DTL in this paper. Admittedly, given that our main example deals with the linearization of the channel model, we do not require here the full power of the logic. Moreover, the security goals that we have considered, secrecy and authentication, constitute safety properties, and therefore we have mostly used formulas with simple past-time operators. Nevertheless, it should be clear that DTL can be used in many other, more complex, scenarios, for example, to model the presence of multiple non-collaborating intruders (possibly with different capabilities) or different channels with different properties (possibly controlled by different intruders). Furthermore, studying other interesting security goals, like forms of fairness in contract-signing protocols, would also require working with liveness properties, for which more complex temporal patterns would be necessary. Finally, note that we have avoided writing formulas with nested communications. This does not mean that such formulas are not necessary, as they result easily from our axioms (for instance, from the composition of axioms \( (P3) \), \( (C2) \), and \( (C1) \)). It is just a consequence of another nice property of DTL: in specifying each agent, locally, in a distributed system, it is possible to replace any DTL formula with nested communications with an equivalent finite set of formulas without nested communications [34]. The application of DTL to such more complex scenarios, in the context of security protocols and web services, is the subject of future work.

Acknowledgments

This work was partially supported by the FP7-ICT-2007-1 Project no. 216471, “AVANTSSAR: Automated Validation of Trust and Security of Service-oriented Architectures”, by the PRIN’07 project “SOFT”, and by FCT and EU FEDER via the KLog project PTDC/MAT/68723/2006 of SQIG-IT and the AMDSC UTAustin/MAT/0057/2008 project of IST. We thank Cas Cremers, Bruno Conchinha, Sebastian Mödersheim, and the anonymous referees for their valuable feedback on earlier drafts of this paper.

References


