Security Theorems via Model Theory*

Joshua D. Guttman
The MITRE Corporation
Worcester Polytechnic Institute

A model-theoretic approach can establish security theorems, which are formulas expressing authentication and non-disclosure properties of protocols. Security theorems have a special form, namely quantified implications $\forall \vec{x}.\,(\phi \supset \exists \vec{y}.\,\psi)$.

Models (interpretations) for these formulas are *skeletons*, partially ordered structures consisting of a number of local protocol behaviors. *Realized* skeletons contain enough local sessions to explain all the behavior, when combined with some possible adversary behaviors.

We show two results. (1) If ϕ is the antecedent of a security goal, then there is a skeleton $\mathbb{A}_\phi$ such that, for every skeleton $\mathbb{B}$, ϕ is satisfied in $\mathbb{B}$ iff there is a homomorphism from $\mathbb{A}_\phi$ to $\mathbb{B}$. (2) A protocol enforces $\forall \vec{x}.\,(\phi \supset \exists \vec{y}.\,\psi)$ iff every realized homomorphic image of $\mathbb{A}_\phi$ satisfies ψ.

Since the program CPSA finds the minimal realized skeletons, or "shapes," that are homomorphic images of $\mathbb{A}_\phi$, if ψ holds in each of these shapes, then the goal holds.

1 Introduction

Much work has been done in recent years on cryptographic protocol analysis. A central problem is, given a protocol, to determine whether a formula, expressing a security goal about its behaviors in the presence of an adversary, is true. If the protocol achieves the goal, one would like some explanation why. If it does not achieve the goal, one would like a counterexample. A security goal is a quantified implication:

$$\forall \vec{x}.\,(\phi_0 \supset \exists \vec{y}.\,\psi). \tag{1}$$

The hypothesis ϕ_0 is a conjunction of atomic formulas describing regular behavior. The conclusion ψ is a disjunction of zero or more such conjunctions, i.e. ψ is $\bigvee_{1 \le i \le k} \phi_i$. When the ϕ_i describe desired behaviors of other regular participants, who are intended to be peers in protocol runs, then this goal is an *authentication* goal. It says that each protocol run contains at least one peer execution from k different possibilities among which the protocol may allow the participants to choose.

When $k = 0$, ψ is the empty disjunction `false`. If ϕ_0 mentions an unwanted disclosure, (1) says the disclosure cannot occur. Hence, security goals with $k = 0$ express *secrecy* goals.[1]

Our models are *skeletons*, partially ordered sets of *regular strands*, i.e. local behaviors of regular participants. A skeleton $\mathbb{A}$ defines a set of executions, namely executions in which images of these strands can be found. We use $\mathbb{A}, \sigma \models \Phi$ in the classical sense, to mean that the formula Φ is satisfied in the skeleton $\mathbb{A}$, when the variable assignment σ determines how variables free in Φ are interpreted.

A skeleton $\mathbb{A}$ is an *execution* if it is *realized*. This means that the message transmissions in $\mathbb{A}$—when combined with possible adversary behavior—suffice to explain every message received in $\mathbb{A}$. A *counterexample* to a goal G is a realized skeleton $\mathbb{C}$ such that, for some variable assignment σ, $\mathbb{C}, \sigma \models \phi$, and for every extension σ' of σ, $\mathbb{C}, \sigma' \models \neg\psi$. $\mathbb{C}$ is a counterexample to G only if $\mathbb{C}$ is realized, even though $\models$ is also well-defined for non-realized skeletons.

*Supported by the MITRE-Sponsored Research program. Author's address: guttman@{mitre.org,wpi.edu}.

[1] We will use ϕ, ϕ_i, etc., for conjunctions of 0 or more atomic formulas, and ψ for disjunctions $\bigvee_{1 \le i \le k} \phi_i$ where $0 \le k$.

We focus on the homomorphisms among models. A *homomorphism* is a structure-preserving map, which may embed one skeleton into a larger one; may identify one strand with another strand that sends and receives similar messages; and may fill in more information about the parameters to the strands. As usual, homomorphisms preserve satisfaction for atomic formulas. Suppose H is a homomorphism $H\colon \mathbb{A} \mapsto \mathbb{B}$. And suppose $\mathbb{A}, \sigma \models \phi$, i.e. the skeleton $\mathbb{A}$ satisfies the atomic formula ϕ under an assignment σ, which maps variables occurring in ϕ to values that may appear in $\mathbb{A}$. Then $\mathbb{B}, H \circ \sigma \models \phi$.

This holds for conjunctions of atomic formulas also. Thus, a security goal $\forall \vec{x}.\, (\phi_0 \supset \exists \vec{y}.\, \psi)$, concerns the homomorphic images of skeletons satisfying ϕ_0. If any homomorphic image $\mathbb{C}$ is realized, then $\mathbb{C}$ should satisfy the disjunction ψ, i.e. $\mathbb{C}$ should satisfy at least one of the disjuncts ϕ_i for $1 \leq i \leq k$.

We already have a method for constructing homomorphisms from a skeleton $\mathbb{A}$ to realized skeletons [8]. The Cryptographic Protocol Shapes Analyzer CPSA is a program that—given a protocol Π and a skeleton of interest $\mathbb{A}$—generates all of the minimal, essentially different realized skeletons that are homomorphic images of $\mathbb{A}$. We call these minimal, essentially different skeletons *shapes*, and there are frequently very few of them.

Main Results. We show how a single run of the search for shapes checks the truth of a security goal. To determine whether Π achieves a goal $G = \forall \vec{x}.\, (\phi_0 \supset \exists \vec{y}.\, \psi)$, we find the shapes for the single skeleton $\mathbb{A}_{\phi_0}$. Two technical results are needed to justified this.

- For any security hypothesis ϕ_0, a single skeleton $\mathbb{A}_{\phi_0}$ characterizes ϕ_0. I.e., for all $\mathbb{B}$:

$$\exists \sigma\,.\ \mathbb{B}, \sigma \models \phi_0 \quad \text{iff} \quad \exists H\,.\ H\colon \mathbb{A}_{\phi_0} \mapsto \mathbb{B}.$$

- There exists a realized $\mathbb{C}$ that is a counterexample to G iff there exists some shape $H\colon \mathbb{A}_\phi \mapsto \mathbb{B}$ where $\mathbb{B}$ provides a counterexample.

Our main results suggest a recipe for evaluating a goal $G = \forall \vec{x}.\, (\phi \supset \exists \vec{y}.\, \psi)$ for a protocol Π.

1. Construct the skeleton $\mathbb{A}_\phi$.
2. Ask CPSA what shapes are accessible in Π, starting from $\mathbb{A}_\phi$.
3. As CPSA delivers shapes, check that each satisfies some disjunct ϕ_i.
4. If the answer is no, this shape is a counterexample to G.
5. If CPSA terminates with no counterexample, then G is achieved.

Since the problem is undecidable [9], it is also possible that CPSA will not terminate. Step 3 is easy, since each ϕ_i is a conjunction of atomic formulas, and each shape is a finite (typically small) structure.

The Language of Goals. For each protocol, we define a first order language $\mathscr{L}(\Pi)$, in which formulas (1) are security goals. $\mathscr{L}(\Pi)$ expresses authentication and secrecy goals [17] for Π, including "injective agreement", as adapted to strand spaces [13].[2] It talks only about the roles. One can say which roles executed, and how far they executed in partial executions, and with what parameters. Saying that different roles executed with the same values for certain parameters is important.

However, $\mathscr{L}(\Pi)$ is carefully designed to limit expressiveness. $\mathscr{L}(\Pi)$ says nothing about the forms of messages, and there are no function symbols for encryption or pairing. The protocol Π determines the forms of messages, so to speak behind $\mathscr{L}(\Pi)$'s back in the semantics. Thus, $\mathscr{L}(\Pi)$ need only stipulate the underlying parameters, when describing what has happened.

[2] $\mathscr{L}(\Pi)$ does not express observational indistinguishability properties, or "strong secrecy" [1].

A benefit of this approach is that related protocols Π, Π' may have similar languages, or indeed identical languages, when corresponding roles use the same parameters to compose messages of different concrete forms. This makes the languages $\mathscr{L}(\Pi)$ suited to analyze protocol transformations (as in [15]) to determine when security goals are preserved. We used them (with inessential differences) in [14], where we gave a syntactic criterion that ensures the safety of combining pairs of protocols. When Π_1, Π_2 meet the criterion, then any goal (1) in $\mathscr{L}(\Pi_1)$ that Π_1 meets is still achieved by $\Pi_1 \cup \Pi_2$. Combining Π_1 with Π_2 to form $\Pi_1 \cup \Pi_2$ is a simple sort of transformation of Π_1.

Some Related Work. To document a protocol meeting a security goal, one might like to provide a proof, e.g. in Paulson's style [19], or in the Protocol Composition Logic [6]. One might also view a counterexample as a syntactic object much like a proof. Symbolic constraint solving techniques (starting with [11, 18, 20]) treat them in this way, using rules, including unification, to construct them. The "adversary-centered" approach of Selinger [21] also leads to a proof-like treatment of protocol counterexamples, and to a model-theoretic view of achieving goals. To show that a goal is met, one exhibits a model in which axioms are satisfied, but the adversary's knowledge does not include any intended secrets. These axioms describe the behavior of the regular (non-compromised) participants. The model is a set that is invariant under disclosures effected by the protocol, but in which the secrets do not appear.

We give another model-theoretic approach to achieving goals, but from a "protocol-centered" point of view rather than an "adversary-centered" one. In contrast to Selinger, who expresses authentication properties of a protocol Π by means of secrecy properties of an expanded protocol Π', we represent secrecy properties as the special case of authentication properties where $k = 0$, as indicated above.

Chein and Mugnier also use homomorphisms to evaluate the truth of implications, e.g. [4, 5], in the context of conceptual graphs. However, that context is quite different, since their *formulas* are graph-like objects, whereas our *interpretations* are graph-like structures. Our framework is tuned to the specific case of cryptographic protocols; for instance, there is no analog of "realized" in their framework.

Structure of this Paper. We start with some examples of protocol goals in a simple protocol that does not achieve all of them, and a corrected protocol that does (Section 2). In Section 3, we define the first order classical languages $\mathscr{L}(\Pi)$ that express security goals for each protocol Π. Section 4 defines skeletons and homomorphisms between them, and gives a semantics for $\mathscr{L}(\Pi)$ using skeletons. We show next in Section 5 that each conjunction ϕ of atomic formulas has a characteristic skeleton. In Section 6, we show how to use the characteristic skeletons to check security goals.

Strand Spaces. A *strand* is a (linearly ordered) sequence of nodes $n_1 \Rightarrow \ldots \Rightarrow n_j$, each of which transmits or receives some message $\mathsf{msg}(n_i)$. A strand may represent the behavior of a principal in a single local session of a protocol, in which case it is a *regular* strand of that protocol, or it may represent a basic adversary activity. Basic adversary activities include receiving a plaintext and a key and transmitting the result of the encryption, and receiving a ciphertext and its matching decryption key, and transmitting the resulting plaintext.

A *protocol* Π is a finite set of strands, which are the *roles* of the protocol. A strand s is an *instance* of a role $\rho \in \Pi$, if $s = \alpha(\rho)$, i.e. if s results from ρ by applying a substitution α to parameters in ρ.

Message t_1 is an *ingredient* of t_2, written $t_1 \sqsubseteq t_2$, if t_1 is used to construct t_2 other than as an encryption key; i.e. $\sqsubseteq$ is the smallest reflexive, transitive relation such that $t_1 \sqsubseteq t_1 \hat{} t_2$, and $t_2 \sqsubseteq t_1 \hat{} t_2$, and $t_1 \sqsubseteq \{|t_1|\}_{t_2}$.

A message t *originates* on a strand node n if (1) $t \sqsubseteq \mathsf{msg}(n)$; (2) n is a transmission node; and (3) $m \Rightarrow^+ n$ implies $t \not\sqsubseteq \mathsf{msg}(m)$. A value that originates only once in an execution is *uniquely originating*,

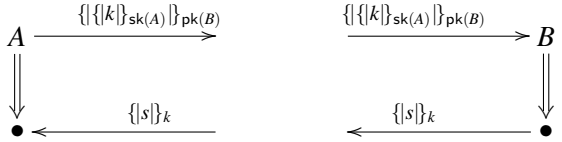


Figure 1: Blanchet's "Simple Example Protocol"

i.e. a freshly chosen value. A value that originates nowhere in an execution may nevertheless be used within regular strands to encrypt or decrypt. However, if the adversary uses a value to encrypt or decrypt, then the key must have been received, and must therefore have originated somewhere. Hence *non-originating* values represent uncompromised long term keys. For more detail, see the Appendix.

2 Examples

Blanchet's Example. We start from an example suggested by Blanchet [2], as shown in Fig. 1. A principal A wishes to have B transmit a secret to A alone. A has a private signature key $\mathsf{sk}(A)$, with a public verification key known to B. B has a private decryption key with a public encryption key $\mathsf{pk}(B)$ known to A. A transmits a freshly chosen symmetric key k to B, signed by A and encrypted for B. B then uses k to encipher the secret s.

Authentication Goals of A. A wants the protocol to ensure that s came from B.

To establish this, we attribute some assumptions to A, from which it may follow that B has transmitted s. A has had a run of her side of the protocol, so the execution will contain the two nodes of the strand shown on the left of the figure. We must assume that B's private decryption key $\mathsf{pk}(B)^{-1}$ is uncompromised, in the sense that this private key never originates. It is thus used only by regular participants in accordance with the protocol, and never by an adversary. Moreover, we assume that the symmetric key k originates only once, namely, on A's first node. In particular, the adversary can not send it until after receiving k. In a word, the adversary will not *guess* k.

$\mathbb{A}_0$ summarizes these assumptions, as shown on the left in Fig. 2. Here $\mathsf{non}_{\mathbb{A}_0}$ is the set of long term keys assumed uncompromised for the sake of this analysis, and $\mathsf{unique}_{\mathbb{A}_0}$ is the set of fresh values assumed uniquely originating. We call a diagram of this kind a *skeleton*.

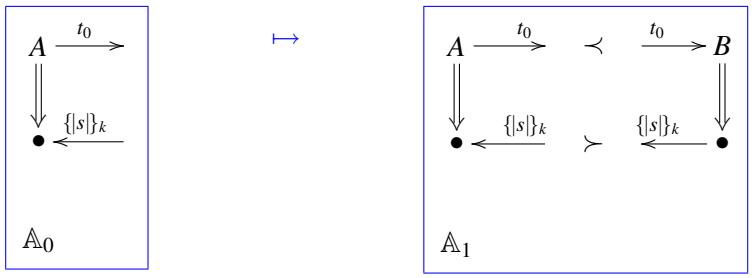
We want now to "analyze" this assumption, by which we mean, to find all shapes accessible from $\mathbb{A}_0$. The shapes give all the minimal, essentially different executions (realized skeletons) accessible from $\mathbb{A}_0$. CPSA reports a single shape, shown on the right of Fig. 2 as $\mathbb{A}_1$. This is what A desired, as B sent s on its transmission node at lower right.

The antecedent of the implication expressing this goal concerns the starting point; is it $\phi^A =$

$$\mathtt{Init2}(\mathtt{n_2},\mathtt{a},\mathtt{b},\mathtt{k},\mathtt{s}) \wedge \mathtt{Unq}(\mathtt{k}) \wedge \mathtt{Non}(\mathtt{sk}(\mathtt{a})) \wedge \mathtt{Non}(\mathtt{inv}(\mathtt{pk}(\mathtt{a}))).$$

The predicate $\mathtt{Init2}(\mathtt{n_2},\mathtt{a},\mathtt{b},\mathtt{k},\mathtt{s})$ says that $\mathtt{n_2}$ is the second node of an initiator strand with the given parameters. The remaining conjuncts express the supplementary assumptions about non-compromised long term keys and a fresh session key. The fact that the only shape we obtain is $\mathbb{A}_1$, which has a responder's second node with the desired parameters, validates:

$$\phi^A \supset \exists \mathtt{m} \,.\, \mathtt{Resp2}(\mathtt{m},\mathtt{a},\mathtt{b},\mathtt{k},\mathtt{s}).$$



$$\{\mathsf{pk}(B)^{-1}\} = \mathsf{non}_{\mathbb{A}_0} = \mathsf{non}_{\mathbb{A}_1}, \{k\} = \mathsf{unique}_{\mathbb{A}_0} = \mathsf{unique}_{\mathbb{A}_1}$$

Figure 2: Skeletons $\mathbb{A}_0, \mathbb{A}_1$, input and output for CPSA, with $t_0 = \{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$

The conclusion of this implication describes some of the additional structure contained in $\mathbb{A}_1$.

Confidentiality of s. The analysis for A's confidentiality goal is similar. We again use the same assumptions, augmented by the pessimistic assumption that s is compromised. We represent this using a trick: We use a *listener node* $\bullet \xleftarrow{s}$ that "hears" the value s shorn of any cryptographic protection. Here we must also assume that $s \in \mathsf{unique}_{\mathbb{A}_2}$, since otherwise possibly the adversary will simply guess (re-originate) s. Thus, we start the analysis with the form shown in Fig. 3. We now reach an impasse: CPSA reports that no shape exists, starting from $\mathbb{A}_2$. Thus, $\mathbb{A}_2$ is *dead*: No realized skeleton can result from it. We express this confidentiality goal in the form:

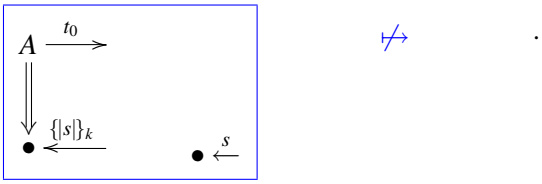
$$\phi^A \wedge \mathtt{Unq}(\mathtt{s}) \wedge \mathtt{Lsn}(\mathtt{m},\mathtt{s}) \supset \mathtt{false}.$$

The conjunct $\mathtt{Lsn}(\mathtt{m},\mathtt{s})$ describes the listener node $\mathtt{m}$ that "hears" the value $\mathtt{s}$. This listener node is incompatible with the other assumptions ϕ^A. Thus, A achieves her goals using this protocol.

Authentication Goal for B. Unfortunately, the situation is less favorable for B. We start the analysis with the skeleton $\mathbb{A}_3$, shown on the left in Fig. 4. It represents the hypothesis ϕ^B:

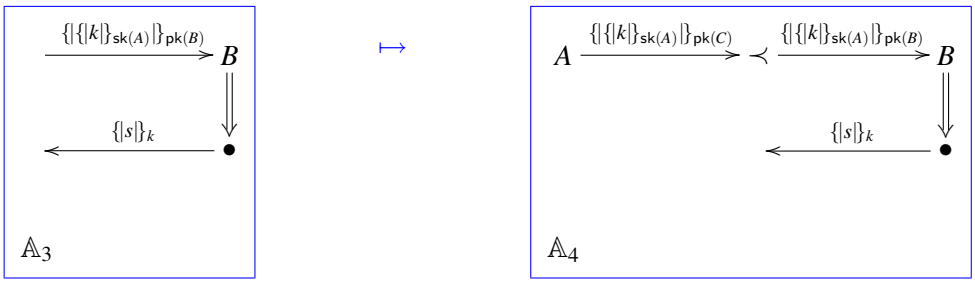
$$\mathtt{Resp2}(\mathtt{m},\mathtt{a},\mathtt{b},\mathtt{k},\mathtt{s}) \wedge \mathtt{Unq}(\mathtt{k}) \wedge \mathtt{Non}(\mathtt{sk}(\mathtt{a})) \wedge \mathtt{Non}(\mathtt{inv}(\mathtt{pk}(\mathtt{a})))$$

We obtain the form $\mathbb{A}_4$ shown on the right in Fig. 4. Unfortunately, we have learnt nothing about the



$$\{\mathsf{pk}(B)^{-1}\} = \mathsf{non}_{\mathbb{A}_0} \qquad \{k,s\} = \mathsf{unique}_{\mathbb{A}_0}$$

Figure 3: $\mathbb{A}_2$, for A's confidentiality analysis, with no shapes



$$\{\mathsf{sk}(A), \mathsf{pk}(B)^{-1}\} = \mathsf{non}_{\mathbb{A}_3} = \mathsf{non}_{\mathbb{A}_4} \quad \{k\} = \mathsf{unique}_{\mathbb{A}_3} = \mathsf{unique}_{\mathbb{A}_4}$$

Figure 4: $\mathbb{A}_3, \mathbb{A}_4$, for B's authentication goal

recipient C for whom A intended this key. Possibly $\mathsf{pk}(C)^{-1}$ is compromised. Thus, the adversary can decrypt $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(C)}$ and use B's public key to construct $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$. Thus, skeleton $\mathbb{A}_4$ is realized, but validates only:

$$\phi^B \supset \exists \mathtt{n}, \mathtt{c} \;.\; \mathtt{Init1}(\mathtt{n}, \mathtt{a}, \mathtt{c}, \mathtt{k}).$$

The expected initiator has got to step 1 of a run with some $\mathtt{c}$, who—the protocol ensures—has originated the key k. However, since possibly $C \neq B$, confidentiality for k and s may fail.

The details of the CPSA run make clear how to fix the protocol. This requires us to replace t_0 with $\{|\{|k\hat{}B|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$, which includes B's identity under A's signature. In fact, Blanchet [2] makes a more complicated suggestion, using the component $\{|k\hat{}A\hat{}B|\}_{\mathsf{sk}(A)}$. However, the CPSA analysis is very precise, and indicates that only the responder's identity needs to be included inside the signature.

3 Language

We now consider the logical representation of our example authentication and confidentiality goals.

Role Predicates. We need to be able to describe the different kinds of nodes that are present in a skeleton, namely the initiator's first and second nodes, and the responder's first and second nodes, each of which has a number of parameters. We must be able to express these parameters, because we know (e.g.) that whether the initiator's first node has B or C as its responder identity parameter makes all the difference. On the other hand, the form of the messages is defined in the protocol, and is thus irrelevant to describing what goals are achieved. Thus, for Blanchet's simple example protocol, we suggest four protocol-specific predicates:

$$\begin{array}{rr} \mathtt{Init1}(\mathtt{m}, \mathtt{a}, \mathtt{b}, \mathtt{k}) & \mathtt{Resp1}(\mathtt{m}, \mathtt{a}, \mathtt{b}, \mathtt{k}) \\ \mathtt{Init2}(\mathtt{m}, \mathtt{a}, \mathtt{b}, \mathtt{k}, \mathtt{s}) & \mathtt{Resp2}(\mathtt{m}, \mathtt{a}, \mathtt{b}, \mathtt{k}, \mathtt{s}) \end{array}$$

Suppose that we are given a variable assignment σ that associates values $\sigma(\mathtt{m}), \sigma(\mathtt{a})$, etc. to the variables $\mathtt{m}, \mathtt{a}$, etc. Then we read $\mathtt{Init1}(\mathtt{m}, \mathtt{a}, \mathtt{b}, \mathtt{k})$ as asserting that $\sigma(\mathtt{m})$ is a node, which is the first step of an initiator strand, and the initiator is $\sigma(\mathtt{a})$, its intended responder is $\sigma(\mathtt{b})$, and it has created the key $\sigma(\mathtt{k})$ for this session. We read $\mathtt{Init2}(\mathtt{m}, \mathtt{a}, \mathtt{b}, \mathtt{k}, \mathtt{s})$ as asserting that $\sigma(\mathtt{m})$ is a node, which is the second step of an initiator strand, and the initiator is $\sigma(\mathtt{a})$, its intended responder is $\sigma(\mathtt{b})$, the session key is $\sigma(\mathtt{k})$, and

the intended secret is $\sigma(\mathtt{s})$. Thus, if $\mathtt{Init2}(\mathtt{m},\mathtt{a},\mathtt{b},\mathtt{k},\mathtt{s})$, then we know that $\mathtt{m}$ is preceded by a node $\mathtt{n}$ such that $\mathtt{Init1}(\mathtt{n},\mathtt{a},\mathtt{b},\mathtt{k})$ is true under the same σ. Analogous explanations hold for the responder role.

These role predicates are similar to those used by Cervesato, Durgin et al. [3, 10] in multiset rewriting.

We also need a role predicate $\mathtt{Lsn}(\mathtt{m},\mathtt{v})$, which says, under an assignment σ, that $\sigma(\mathtt{m})$ is a listener node that receives that value $\sigma(\mathtt{v})$. The language $\mathscr{L}(\Pi)$, where Π is Blanchet's example protocol, contains these five role predicates.

Shared Vocabulary. All languages $\mathscr{L}(\Pi)$ also contain some additional predicates. $\mathtt{Preceq}(\mathtt{m},\mathtt{n})$ expresses the causal partial ordering $\preceq$. $\mathtt{Col}(\mathtt{m},\mathtt{n})$ says that $\sigma(\mathtt{m})$ and $\sigma(\mathtt{n})$ are collinear, i.e. they lie on the same strand. $\mathtt{Non}(\mathtt{v})$ and $\mathtt{Unq}(\mathtt{v})$ express the assumptions that $\sigma(\mathtt{v}) \in \mathsf{non}$ and $\sigma(\mathtt{v}) \in \mathsf{unique}$, resp.

In Section 2, we also used the function symbols $\mathtt{sk},\mathtt{pk}$, and $\mathtt{inv}$ to talk about the long term keys signature keys of principals, their long term public encryption keys, and the inverses of those keys.

This is the full language $\mathscr{L}(\Pi)$ where Π is Blanchet's example protocol. As it happens, the language $\mathscr{L}(\Pi')$, for the corrected version of the protocol, is identical. The roles are the same, each with the same number of nodes, and with the same parameters. Thus, nothing in the language needs to change.

Apparently, for every goal $G \in \mathscr{L}(\Pi)$, if G is achieved in Blanchet's example protocol Π, then G is also achieved in its correction Π'. Moreover, additional formulas are achieved in Π'. A satisfactory theory of protocol transformation should give ways to prove (or disprove) intuitions like this one.

The Languages $\mathscr{L}(\Pi)$. For each protocol Π, $\mathscr{L}(\Pi)$ is a language for talking about its executions. We use typewriter font $\mathtt{x},\mathtt{m}$, etc. for syntactic items such as variables or predicates within the language.

Suppose that Π has r protocol-specific roles $\{\rho_1,\ldots,\rho_r\}$, where each role ρ_i is of length $|\rho_i|$, and the listener role. We let $\{\mathtt{RP}_{0,1},\ldots,\mathtt{RP}_{r,|\rho_r|}\}$ be a collection of $1+\sum_i|\rho_i|$ predicate symbols. $\mathtt{RP}_{0,1}$ is the listener role predicate, which we will write as $\mathtt{Lsn}(\mathtt{m},\mathtt{v})$, indicating that node $\mathtt{m}$ receives the value $\mathtt{v}$. Each remaining predicate $\mathtt{RP}_{ij}$ takes parameters $(\mathtt{m},\mathtt{v}_1,\ldots,\mathtt{v}_k)$ where the j^{th} node on role ρ_i, and its predecessors, have involved k parameters.

We write $\mathsf{fv}(\Phi),\mathsf{bv}(\Phi)$ for the free and bound variables of any formula Φ, defined in the usual way. The empty disjunction $\bigvee_{i\in\emptyset}\phi_i$ is identical with $\mathtt{false}$; a one-element disjunction or conjunction is identical with its single disjunct or conjunct.

Definition 3.1 *1. $\mathscr{L}(\Pi)$ is the classical first order quantified language with vocabulary:*

Variables *(unsorted) ranging over messages and nodes;*

Function symbols $\mathtt{sk}$, $\mathtt{pk}$, $\mathtt{inv}$;[3]

Predicate symbols *equality* $\mathtt{u}=\mathtt{v}$, *falsehood* $\mathtt{false}$ *(no arguments), and:*

- $\mathtt{Non}(\mathtt{v})$, *and* $\mathtt{Unq}(\mathtt{v})$;
- $\mathtt{Col}(\mathtt{m},\mathtt{n})$ *and* $\mathtt{Preceq}(\mathtt{m},\mathtt{n})$;
- *One role predicate* $\mathtt{RP}_{ij}$ *for each* $\rho_i \in \Pi$ *and* j *with* $1 \le j \le |\rho_i|$.
  The predicate $\mathtt{RP}_{ij}(\mathtt{m},\mathtt{v}_1,\ldots,\mathtt{v}_k)$ *for the* j^{th} *node on* ρ_i *has as arguments: a variable* $\mathtt{m}$ *for the node, and variables* $\mathtt{v}_\ell$ *for each of the* k *parameters that have appeared in any of* ρ_i*'s first* j *messages.*

2. A security claim ϕ *is a conjunction of atomic formulas of $\mathscr{L}(\Pi)$ such that two conditions hold:*

(a) Any two role predicate conjuncts have different variables as their first arguments $\mathtt{n},\mathtt{n}'$.

(b) If a conjunct is not a role predicate, then each variable or key term that appears as an argument to it also appears as argument to some role predicate $\mathtt{RP}_{ij}(\mathtt{n},\mathtt{t}_1,\ldots,\mathtt{t}_i)$.

[3] We call terms built with these unary function symbols *key terms*.

3. *A sentence* $\forall \vec{x}.(\phi \supset \exists \vec{y}.\psi)$ *is a* security goal *if (1) the* $\vec{x}$*s and* $\vec{y}$*s are disjoint; (2)* ϕ *is a security claim; and (3)* ψ *is a disjunction* $\bigvee_i \phi_i$ *of conjunctions* ϕ_i *of atomic formulas.*

The conditions in Clauses 2a–2b on security claims ϕ allow us to construct a single skeleton $\mathbb{A}_\phi$ to characterize ϕ (Thm. 5.2). Without Clause 2a, we would have to be rather careful in our choice of message algebras and protocols to ensure that there is a single "most general" role that applies when two roles can have common instances. The role predicate $\mathtt{RP}_{ij}$ in Clause 2b serves as an implicit sort declaration for the variables $\mathtt{x}$ appearing in it. The sorts of parameters within the role ρ_i determines the set of values that may lead to true instances of this atomic formula.

We have already illustrated three security goals earlier, in Section 2; or, more precisely, we have shown three formulas whose universal closures are security goals. Another relevant example is the "missing" confidentiality goal that a responder would have wanted, but was actually not achieved by Π but only by its correction Π'. It is the universal closure of:

$$\mathtt{Resp2}(\mathtt{n_2},\mathtt{a},\mathtt{b},\mathtt{k},\mathtt{s}) \wedge \mathtt{Unq}(\mathtt{k}) \wedge \mathtt{Non}(\mathtt{sk}(\mathtt{a})) \wedge \mathtt{Non}(\mathtt{inv}(\mathtt{pk}(\mathtt{a}))) \wedge \mathtt{Lsn}(\mathtt{m},\mathtt{s}) \supset \mathtt{false}.$$

Axiomatizing Protocols. $\mathscr{L}(\Pi)$ is specifically intended to limit expressiveness, and there is no way to axiomatize the behavior of protocols within it. However, the slightly larger language $\mathscr{L}^+(\Pi)$ appears sufficient to axiomatize protocol behaviors, and derive security goals. It adds to $\mathscr{L}(\Pi)$:

Function symbols $\mathtt{concat}(\mathtt{v_1},\mathtt{v_2})$ and $\mathtt{enc}(\mathtt{v_1},\mathtt{v_2})$, representing the concatenation of two messages v_1, v_2 and the encryption of a message v_1 using a second message v_2 as key; and $\mathtt{msgAt}(\mathtt{n_1})$ returning the message transmitted or received on the node n_1;

Predicate symbols $\mathtt{Xmit}(\mathtt{n_1})$ and $\mathtt{Rcv}(\mathtt{n_1})$, true if n_1 is a transmission node or reception node, resp.

A few inductively defined notions such as "message t_0 is found only within the set of encryptions S in message t_1" [8, extended version, Def. 6] must be introduced using these primitives. The property of a skeleton being realized can then be expressed as a closed sentence. With these notions, the reasoning encoded in CPSA could be carried out axiomatically, at least in theory, within $\mathscr{L}^+(\Pi)$. The important theorems would be the security goals, which lie within the sublanguage $\mathscr{L}(\Pi)$.

4 Skeletons, Homomorphisms, and Satisfaction

Before we define the satisfaction relation $\mathbb{A}, \sigma \models \phi$, we must define the skeletons that we have already worked with in Section 2. We start by summarizing our assumptions about the message algebra; more detail may be found in Appendix A.

Message Algebra. Let $\mathfrak{A}_0$ be an algebra equipped with some operators and a set of homomorphisms $\eta : \mathfrak{A}_0 \to \mathfrak{A}_0$. We call members of $\mathfrak{A}_0$ *atoms*.

For the sake of definiteness, we will assume here that $\mathfrak{A}_0$ is the disjoint union of infinite sets of *nonces*, *atomic keys*, *names*, and *texts*. The operator $\mathtt{sk}(a)$ maps names to (atomic) signature keys, and K^{-1} maps an asymmetric atomic key to its inverse, and a symmetric atomic key to itself. Homomorphisms η are maps that respect sorts, and act homomorphically on $\mathtt{sk}(a)$ and K^{-1}.

Let X is an infinite set disjoint from $\mathfrak{A}_0$; its members—called *indeterminates*—act like unsorted variables. $\mathfrak{A}$ is freely generated from $\mathfrak{A}_0 \cup X$ by two operations: encryption $\{|t_0|\}_{t_1}$ and tagged concatenation *tag* $t_0{}^\frown t_1$, where the tags *tag* are drawn from some set *TAG*. For a distinguished tag *nil*, we write

nil $t_0{}^\frown t_1$ as $t_0{}^\frown t_1$ with no tag. In $\{|t_0|\}_{t_1}$, a non-atomic key t_1 is a symmetric key. Members of $\mathfrak{A}$ are called *messages*.

A homomorphism $\alpha = (\eta, \chi)\colon \mathfrak{A} \to \mathfrak{A}$ consists of a homomorphism η on atoms and a function $\chi\colon X \to \mathfrak{A}$. It is defined for all $t \in \mathfrak{A}$ by the conditions:

$$\alpha(a) = \eta(a), \quad \text{if } a \in \mathfrak{A}_0 \qquad \alpha(\{|t_0|\}_{t_1}) = \{|\alpha(t_0)|\}_{\alpha(t_1)}$$
$$\alpha(x) = \chi(x), \quad \text{if } x \in X \qquad \alpha(tag\ t_0{}^\frown t_1) = tag\ \alpha(t_0){}^\frown \alpha(t_1)$$

Thus, atoms serve as typed variables, replaceable only by other values of the same sort, while indeterminates x are untyped. Indeterminates x serve as blank slots, to be filled by any $\chi(x) \in \mathfrak{A}$. Indeterminates and atoms are jointly *parameters*.

This $\mathfrak{A}$ has the most general unifier property, which we will rely on. That is, suppose that for $v, w \in \mathfrak{A}$, there exist α, β such that $\alpha(v) = \beta(w)$. Then there are α_0, β_0, such that $\alpha_0(v) = \beta_0(w)$, and whenever $\alpha(v) = \beta(w)$, then α and β are of the forms $\gamma \circ \alpha_0$ and $\gamma \circ \beta_0$.

Skeletons. A skeleton is a partially ordered set of nodes, together with assumptions non about uncompromised long term keys and unique about freshly chosen values. We write $s \downarrow i$ for the i^{th} node along s, using 1-based indexing.

A *skeleton* $\mathbb{A}$ consists of (possibly partially executed) role instances, i.e. a finite set of nodes, $\mathsf{nodes}(\mathbb{A})$, with two additional kinds of information:

1. A partial ordering $\preceq_{\mathbb{A}}$ on $\mathsf{nodes}(\mathbb{A})$;
2. Sets $\mathsf{unique}_{\mathbb{A}}, \mathsf{non}_{\mathbb{A}}$ of atomic values assumed uniquely originating and non-originating in $\mathbb{A}$.

$\mathsf{nodes}(\mathbb{A})$ and $\preceq_{\mathbb{A}}$ must respect the strand order, i.e. if $n_1 \in \mathsf{nodes}(\mathbb{A})$ and $n_0 \Rightarrow n_1$, then $n_0 \in \mathsf{nodes}(\mathbb{A})$ and $n_0 \preceq_{\mathbb{A}} n_1$. If $a \in \mathsf{unique}_{\mathbb{A}}$, then a must originate at most once in $\mathsf{nodes}(\mathbb{A})$. If $a \in \mathsf{non}_{\mathbb{A}}$, then a must originate nowhere in $\mathsf{nodes}(\mathbb{A})$, though a or a^{-1} may be the key encrypting some ingredient of $n \in \mathsf{nodes}(\mathbb{A})$.

$\mathbb{A}$ is a *preskeleton* if it meets the conditions except that some values $a \in \mathsf{unique}_{\mathbb{A}}$ may originate more than once in $\mathsf{nodes}(\mathbb{A})$. If $\mathbb{A}$ is a preskeleton, and it is possible to extract a skeleton by identifying nodes and atoms, then there is a canonical, most general way to do so [8, extended version, Prop. 6]. The canonical skeleton extracted from $\mathbb{A}$ is called the *hull* of $\mathbb{A}$. We write $\mathsf{hull}_{\mathbb{A}}$ for the homomorphism (Def. 4.1) that maps a preskeleton $\mathbb{A}$ to its hull.

A skeleton $\mathbb{A}$ is a *skeleton for* a protocol Π if all of its strands are strands of Π.

$\mathbb{A}$ is *realized* if it can occur without additional activity of *regular* participants; i.e., for every reception node n, the adversary can construct $\mathsf{msg}(n)$ via the Dolev-Yao adversary actions,[4] using as inputs:

1. the messages $\mathsf{msg}(m)$ where $m \prec_{\mathbb{A}} n$ and m is a transmission node;
2. indeterminates x; and
3. any atomic values a such that $a \notin (\mathsf{non}_{\mathbb{A}} \cup \mathsf{unique}_{\mathbb{A}})$, or such that $a \in \mathsf{unique}_{\mathbb{A}}$ but a originates nowhere in $\mathbb{A}$.

Definition 4.1 *Let $\mathbb{A}_0, \mathbb{A}_1$ be preskeletons, α a homomorphism on $\mathfrak{A}$, and $\zeta\colon \mathsf{nodes}_{\mathbb{A}_0} \to \mathsf{nodes}_{\mathbb{A}_1}$. $H = [\zeta, \alpha]$ is a* (skeleton) homomorphism *if*

1*a*. *For all $n \in \mathbb{A}_0$, $\mathsf{msg}(\zeta(n)) = \alpha(\mathsf{msg}(n))$, with the same direction, either transmission or reception;*

1*b*. *For all s, i, if $s \downarrow i \in \mathbb{A}$, then there is an s' s.t. for all $j \leq i$, $\quad \zeta(s \downarrow j) = s' \downarrow j$;*

[4] The Dolev-Yao adversary actions are: concatenating messages and separating the pieces of a concatenation; encrypting a given plaintext using a given key; and decrypting a given ciphertext using the matching decryption key.

2. $n \preceq_{\mathbb{A}_0} m$ *implies* $\zeta(n) \preceq_{\mathbb{A}_1} \zeta(m)$;
3. $\alpha(\mathsf{non}_{\mathbb{A}_0}) \subseteq \mathsf{non}_{\mathbb{A}_1}$;

4*a*. $\alpha(\mathsf{unique}_{\mathbb{A}_0}) \subseteq \mathsf{unique}_{\mathbb{A}_1}$;

4*b*. *If* $a \in \mathsf{unique}_{\mathbb{A}_0}$ *and* a *originates at* $n \in \mathsf{nodes}_{\mathbb{A}_0}$, *then* $\alpha(a)$ *originates at* $\zeta(n) \in \mathsf{nodes}_{\mathbb{A}_1}$.

We write $H\colon \mathbb{A}_0 \mapsto \mathbb{A}_1$ *when* H *is a homomorphism from* $\mathbb{A}_0$ *to* $\mathbb{A}_1$. *When* $\alpha(a) = \alpha(a)'$ *for every* a *that is an ingredient or is used for encryption in* $\mathsf{dom}(\zeta)$, *then* $[\zeta,\alpha] = [\zeta,\alpha']$; *i.e.,* $[\zeta,\alpha]$ *is the equivalence class of pairs under this relation.*

The condition for $[\zeta,\alpha] = [\zeta,\alpha']$ implies that the action of α on atoms not mentioned in the $\mathbb{A}_0$ is irrelevant. We write $H(n)$ for $\zeta(n)$ or $H(a)$ for $\alpha(a)$, when $H = [\zeta,\alpha]$. Evidently, preskeletons and homomorphisms form a category, of which skeletons and homomorphisms are subcategory.

In Section 2, we have already given examples of homomorphisms. Each of the shapes we have considered is a homomorphic image of its starting point. Thus, for instance, we have homomorphisms $\mathbb{A}_0 \mapsto \mathbb{A}_1$ and $\mathbb{A}_3 \mapsto \mathbb{A}_4$. $\mathbb{A}_2$ is dead in the sense that there is no realized $\mathbb{B}$ such that $\mathbb{A}_2 \mapsto \mathbb{B}$.

Our first CPSA run in Section 2 tells us that every homomorphism from $\mathbb{A}_0$ to a realized skeleton goes "by way of" $\mathbb{A}_1$ [8]. That is, if $\mathbb{B}$ is realized and $H\colon \mathbb{A}_0 \mapsto \mathbb{B}$, then $H = H_1 \circ H_0$ where $H_0\colon \mathbb{A}_0 \mapsto \mathbb{A}_1$. Thus, any realized skeleton accessible from $\mathbb{A}_0$ has at least the structure contained in $\mathbb{A}_1$, homomorphisms being structure-preserving maps.

In Figs. 2 and 4, the homomorphism simply adds nodes. I.e. ζ is an embedding and α is the identity. However, in other homomorphisms, ζ may be a bijection and α does the work, mapping distinct values to the same result. In other cases ζ is non-injective, mapping two distinct strands in source to the same strand in the target. For instance, suppose that $\mathbb{A}$ is a preskeleton but not a skeleton, because some $a \in \mathsf{unique}_{\mathbb{A}}$ originates on two strands. If the map $\mathsf{hull}_{\mathbb{A}}$ is well defined, then $\mathsf{hull}_{\mathbb{A}}$ must map both strands on which a originates to the same strand in the target skeleton. Hence, non-trivial $\mathsf{hull}_{\mathbb{A}}$ maps are examples of non-injective homomorphisms.

Semantics for $\mathscr{L}(\Pi)$. The semantics for $\mathscr{L}(\Pi)$ are classical, with each structure a skeleton for the protocol Π. This requirement builds the permissible behaviors of Π directly into the semantics without requiring an explicit axiomatization.

An *assignment* σ for $\mathbb{A}$ is a partial function from variables of $\mathscr{L}(\Pi)$ to $\mathfrak{A} \cup \mathsf{nodes}(\Pi)$. By convention, if σ is undefined for any variable $\mathtt{x}$ in $\mathsf{fv}(\Psi)$, then $\mathbb{A},\sigma \not\models \Psi$. We write $\sigma_1 \oplus \sigma_2$ for the partial function that—for σ_1 and σ_2 with disjoint domains—acts as either σ_i on the domain of that σ_i.

Definition 4.2 *Let* $\mathbb{A}$ *be a skeleton for* Π. *Extend any assignment* σ *to key terms of* $\mathscr{L}(\Pi)$ *via the rules:* $\sigma(\mathtt{sk}(\mathtt{t})) = \mathsf{sk}(\sigma(\mathtt{t}))$, $\sigma(\mathtt{inv}(\mathtt{t})) = (\sigma(\mathtt{t}))^{-1}$.

Satisfaction. $\mathbb{A},\sigma \models \Phi$ *is defined via the standard Tarski inductive clauses for the classical first order logical constants, and the base clauses:*

$\mathbb{A},\sigma \models \mathtt{u} = \mathtt{v}$ *iff* $\sigma(\mathtt{u}) = \sigma(\mathtt{v})$;
$\mathbb{A},\sigma \models \mathtt{Non}(\mathtt{v})$ *iff* $\sigma(\mathtt{v}) \in \mathsf{non}_{\mathbb{A}}$;
$\mathbb{A},\sigma \models \mathtt{Unq}(\mathtt{v})$ *iff* $\sigma(\mathtt{v}) \in \mathsf{unique}_{\mathbb{A}}$;
$\mathbb{A},\sigma \models \mathtt{Col}(\mathtt{m},\mathtt{n})$ *iff* $\sigma(\mathtt{m}),\sigma(\mathtt{n}) \in \mathsf{nodes}(\mathbb{A})$, *and either* $\sigma(\mathtt{m}) \Rightarrow^* \sigma(\mathtt{n})$ *or* $\sigma(\mathtt{n}) \Rightarrow^* \sigma(\mathtt{m})$;
$\mathbb{A},\sigma \models \mathtt{Preceq}(\mathtt{m},\mathtt{n})$ *iff* $\sigma(\mathtt{m}) \preceq_{\mathbb{A}} \sigma(\mathtt{n})$;

and, for each role $\rho_i \in \Pi$ *and index* j *on* ρ_i, *the predicate* $\mathtt{RP}_{ij}(\mathtt{m},\mathtt{v}_1,\ldots,\mathtt{v}_k)$ *obeys the clause*

$\mathbb{A},\sigma \models \mathtt{RP}_{ij}(\mathtt{m},\mathtt{v}_1,\ldots,\mathtt{v}_k)$ *iff* $\sigma(\mathtt{m}) \in \mathsf{nodes}(\mathbb{A})$, *and*

$\sigma(\mathtt{m})$ *is an instance of the* j^{th} *node on role* ρ_i,
with the parameters $\sigma(\mathtt{v}_1), \ldots, \sigma(\mathtt{v}_k)$.

We write $\mathbb{A} \models \Phi$ *when* $\mathbb{A}, \sigma \models \Phi$ *for all* σ, *e.g. when* Φ *is a sentence satisfied by* $\mathbb{A}$.

In protocols where there are two different roles ρ_i, ρ_h that differ only after their first j nodes—typically, because they represent different choices at a branch point after the j^{th} node [16, 12]—the two predicates $\mathtt{RP}_{ij}$ and $\mathtt{RP}_{hj}$ are equivalent, as Def. A.1 makes precise.

Lemma 4.3 *Suppose* ϕ *is an atomic formula, and* $H\colon \mathbb{A} \mapsto \mathbb{B}$. *If* $\mathbb{A}, \sigma \models \phi$, *then* $\mathbb{B}, H \circ \sigma \models \phi$.

5 Characteristic Skeletons

We write $\sigma \restriction \mathsf{fv}(\Phi)$ for the partial function σ restricted in domain to the free variables of Φ.

Definition 5.1 *A pair* $\mathbb{A}, \sigma_*$ *is* characteristic *for a formula* Φ *iff* $\mathbb{A}, \sigma_* \models \Phi$ *and, for all* $\mathbb{B}, \sigma$,

$$\mathbb{B}, \sigma \models \Phi \quad \textit{implies} \quad \exists! H\,.\, H\colon \mathbb{A} \mapsto \mathbb{B} \ \textit{and} \ \sigma \restriction \mathsf{fv}(\Phi) = H \circ \sigma_*. \tag{2}$$

If there is such a σ_*, *then* $\mathbb{A}$ *is a* characteristic skeleton *for* Φ.

Being a homomorphic image of this $\mathbb{A}$ characterizes satisfiability of Φ. $\mathbb{A}$ has minimal structure needed to make Φ true, in the sense that Φ is satisfiable in any $\mathbb{A}'$ just in case $\mathbb{A}'$ results from $\mathbb{A}$ by a structure-preserving map (a homomorphism). From the form of the definition, $\mathbb{A}, \sigma_*$ is universal among interpretations satisfying Φ, and such a $\mathbb{A}, \sigma_*$ will be unique to within isomorphism.

Constructing a Characteristic Skeleton. In order to construct a characteristic skeleton $\mathsf{cs}(\phi)$ for a security claim $\phi = \bigwedge_{1 \le i \le \ell} \phi_i$, we treat the successive atomic formulas ϕ_j in turn. As we do so, we maintain two data structures. One is an assignment σ which summarizes what atomic value or node we have associated to each variable we have seen so far. Initially σ is the empty function. The other is the characteristic skeleton $\mathsf{cs}(\bigwedge_{1 \le i \le j-1} \phi_i)$ constructed from the part of the formula seen so far. This is initially the empty skeleton. If ϕ is unsatisfiable, then instead of returning $\mathsf{cs}(\phi), \sigma$ we must fail.

We assume that the conjuncts of ϕ have been reordered if necessary so that atomic formulas containing role predicates precede atomic formulas of the other forms. For convenience, we also eliminate equations by replacing the left hand side by the right hand side throughout the remainder of the formula.

Base Case. If $\ell = 0$, so that $\phi = \bigwedge_{1 \le i \le 0} \phi_i = \mathtt{true}$, then let $\mathsf{cs}(\phi)$ be the empty skeleton, and let σ_0 be the empty (nowhere defined) substitution.

Recursive Step. Let $\phi = \bigwedge_{1 \le i \le \ell+1} \phi_i$, and let $\mathbb{A}_\ell = \mathsf{cs}(\bigwedge_{1 \le i \le \ell} \phi_i)$ be a characteristic skeleton for all but the last conjunct, relative to σ_ℓ. We take cases on the form of the last conjunct, $\phi_{\ell+1}$:

$\mathtt{RP}_{ij}(\mathtt{m}, \mathtt{t}_1, \ldots, \mathtt{t}_k)$**:** By clause 2a in Defn. 3.1, the variable $\mathtt{m}$ is not in the domain of σ_ℓ. If variables appearing in $\mathtt{t}_1, \ldots, \mathtt{t}_k$, are not in the domain of σ_ℓ, select atoms of appropriate sorts, not yet appearing in $\mathbb{A}_\ell$, letting σ' be the result of extending σ_ℓ with these choices.
Let $n_1 \Rightarrow \ldots \Rightarrow n_j$ be the first j nodes of the role ρ_i, instantiated with the values $\sigma'(\mathtt{t}_1), \ldots, \sigma'(\mathtt{t}_k)$. If any n_λ (with $1 \le \lambda \le j$) originates any value $a \in \mathsf{non}_{\sigma_\ell}$, then we must fail.
Otherwise, let the preskeleton $\mathbb{A}'$ be the result of adding $n_1 \Rightarrow \ldots \Rightarrow n_j$ to $\mathbb{A}_\ell$, and let $\mathbb{A}_{\ell+1}$ be its hull. If the hull is undefined, fail. Otherwise, define $\sigma_{\ell+1} = (\mathsf{hull}_{\mathbb{A}'} \circ \sigma') \oplus (\mathtt{m} \mapsto n_j)$. Return $\mathbb{A}_{\ell+1}, \sigma_{\ell+1}$.

$\mathtt{Non}(\mathtt{t})$: By clause 2b in Defn. 3.1, $\sigma_\ell(\mathtt{t}) = v$ is well defined. If the result of adding v to $\mathsf{non}_{\mathbb{A}_\ell}$ is a skeleton, then this skeleton is $\mathbb{A}_{\ell+1}$. Otherwise, we fail. Let $\sigma_{\ell+1} = \sigma_\ell$.

$\mathtt{Unq}(\mathtt{t})$: By clause 2b in Defn. 3.1, $\sigma_\ell(\mathtt{t}) = v$ is well defined. If the result of adding v to $\mathsf{unique}_{\mathbb{A}_\ell}$ is a preskeleton $\mathbb{A}'$ whose hull is a well-defined skeleton, then this skeleton is $\mathbb{A}_{\ell+1}$. Otherwise, we fail. Let $\sigma_{\ell+1} = \mathsf{hull}_{\mathbb{A}'} \circ \sigma_\ell$.

$\mathtt{Preceq}(\mathtt{m},\mathtt{n})$: By clause 2b in Defn. 3.1, $\sigma_\ell(\mathtt{m})$ and $\sigma_\ell(\mathtt{n})$ are well defined. Let $\mathbb{A}_{\ell+1}$ be $\mathbb{A}_\ell$ with the ordering enriched so that $\sigma_\ell(\mathtt{m}) \preceq_{\mathbb{A}_{\ell+1}} \sigma_\ell(\mathtt{n})$, failing if the latter introduces a cycle because in fact $\sigma_\ell(\mathtt{n}) \preceq_{\mathbb{A}_\ell} \sigma_\ell(\mathtt{m})$. Let $\sigma_{\ell+1} = \sigma_\ell$.

$\mathtt{Col}(\mathtt{m},\mathtt{n})$: By clause 2b in Defn. 3.1, $\sigma_\ell(\mathtt{m}) = s \downarrow k$ and $\sigma_\ell(\mathtt{n}) = s' \downarrow k'$ are well defined. If one strand, e.g. s, is at least as long as the other, we would like to map the successive nodes of s' to nodes of s. However, their messages and directions in $\mathbb{A}_\ell$ may not be the same. If the directions (transmit vs. receive) conflict, then we must fail. Otherwise, if the successive messages are unequal, we may succeed by unifying them.

Let β be the most general unifier such that, for each i where both $s \downarrow i$ and $s' \downarrow i$ are defined,

$$\beta(\mathsf{msg}(s \downarrow i)) = \beta(\mathsf{msg}(s' \downarrow i)).$$

Let $\mathbb{A}'$ be the preskeleton resulting from applying β throughout $\mathbb{A}$, failing if this is impossible because any value in $\mathsf{non}_{\mathbb{A}'}$ would originate somewhere. Let $\mathbb{A}''$ be the preskeleton resulting from omitting $\beta(s')$, and identifying its nodes with those of $\beta(s)$, failing if this identification introduces any cycle into the ordering. If $\mathbb{A}''$ does not have a well defined hull, then fail. Otherwise, let that hull be $\mathbb{A}_{\ell+1}$. Let $\sigma_{\ell+1} = \mathsf{hull}_{\mathbb{A}''} \circ \beta \circ \sigma_\ell$.

Theorem 5.2 *If a security claim $\phi = \bigwedge_{1 \le i \le \ell} \phi_i$ is unsatisfiable, the procedure above fails. If ϕ is satisfiable, then the procedure returns a pair $\mathbb{A}, \sigma$ that is characteristic for ϕ.*

Proof: We follow the inductive definition of $\mathsf{cs}(\phi)$.

Base Case. Let $\ell = 0$, and $\phi = \mathtt{true}$. Then $\mathsf{cs}(\mathtt{true})$ is the empty skeleton $\mathbb{A}_0$. In fact, every $\mathbb{B}$ satisfies $\mathtt{true}$ via the empty substitution, and there exists exactly one homomorphism $H\colon \mathbb{A}_0 \mapsto \mathbb{B}$.

Recursive Step. Let $\phi = \bigwedge_{1 \le i \le \ell+1} \phi_i$, and let $\phi^- = \bigwedge_{1 \le i \le \ell} \phi_i$ be its predecessor, with all but the last conjunct of ϕ. Let $\mathbb{A}_\ell = \mathsf{cs}(\phi^-)$ be a characteristic skeleton for all but the last conjunct, relative to σ_ℓ. In particular, ϕ^- is satisfiable. If ϕ is unsatisfiable, then we need to check we will fail at this step. If this step succeeds, then we need to show that $\mathbb{A}_{\ell+1}, \sigma_{\ell+1}$ are characteristic for $\mathsf{cs}(\phi)$.

Suppose that, for any $\mathbb{B}, \tau$, we have $\mathbb{B}, \tau \models \phi$. Then $\mathbb{B}$ also satisfies ϕ^-, so by the induction hypothesis, there is a unique homomorphism $H_\ell = [\zeta_\ell, \alpha_\ell]\colon \mathbb{A}_\ell \mapsto \mathbb{B}$ to within isomorphism. Moreover, $\tau \restriction \mathsf{fv}(\phi^-) = \alpha_\ell \circ \sigma_\ell$. We take cases on the form of the last conjunct, $\phi_{\ell+1}$. In each case, H_ℓ can be adjusted to form a unique homomorphism $H_{\ell+1}\colon \mathbb{A}_{\ell+1} \mapsto \mathbb{B}$, to within isomorphism.

We use the same notation as in the corresponding cases of the definition of cs.

$\mathtt{RP}_{ij}(\mathtt{m},\mathtt{t}_1,\ldots,\mathtt{t}_k)$: This is not jointly satisfiable with ϕ^- iff the new nodes $n_1 \Rightarrow \ldots \Rightarrow n_j$ originate a value $a \in \mathsf{non}_{\mathbb{A}_\ell}$, or if the hull is not defined. In these cases, cs fails.

$\mathbb{B}, \tau \models \phi$, and, since $\mathtt{RP}_{ij}(\mathtt{m},\mathtt{t}_1,\ldots,\mathtt{t}_k)$ is its last conjunct $\mathbb{B}, \tau \models \mathtt{RP}_{ij}(\mathtt{m},\mathtt{t}_1,\ldots,\mathtt{t}_k)$.

Thus, $\tau(\mathtt{m})$ is an instance of $\rho_i \downarrow j$ with parameters $\tau(\mathtt{t}_1),\ldots,\tau(\mathtt{t}_k)$. Naming $\tau(\mathtt{m}) = n'_j$, we have $n'_1 \Rightarrow \ldots \Rightarrow n'_j$, and by the construction of $\mathsf{cs}(\phi)$, we have $n_1 \Rightarrow \ldots \Rightarrow n_j$ in $\mathsf{cs}(\phi)$. Thus, we can extend the node map ζ_ℓ to $\zeta_{\ell+1}$ by mapping each n_λ to n'_λ. This is the only extension compatible with τ.

Moreover, for each new variable v appearing in $\mathtt{t}_1,\ldots,\mathtt{t}_k$, we extend α_ℓ by sending $\sigma_{\ell+1}(\mathtt{v})$ to $\tau(\mathtt{v})$. By the construction of $\sigma_{\ell+1}(\mathtt{v})$, this is a new value, not equal to any value mentioned in $\mathbb{A}_\ell$. So $\sigma_{\ell+1}$ is a partial function. Moreover, there is no other way to extend σ_ℓ compatible with τ. The resulting $\alpha_{\ell+1}$, together with $\zeta_{\ell+1}$, forms a homomorphism $\mathbb{A}_{\ell_1} \mapsto \mathbb{B}$, and is uniquely determined.

$\mathtt{Non}(\mathtt{t})$**:** If not jointly satisfiable with ϕ^-, then $\sigma_{\ell+1}(\mathtt{t})$ originates in $\mathbb{A}_\ell$, and cs fails.
Since $\mathbb{B}, \tau \models \phi_{\ell+1}$, $\tau(\mathtt{t}) \in \mathsf{non}_{\mathbb{B}}$, so H_ℓ is also a homomorphism from $\mathbb{A}_{\ell+1}$ to $\mathbb{B}$.

$\mathtt{Unq}(\mathtt{t})$**:** The universality of the $\mathsf{hull}_{\mathbb{A}'}$ homomorphism among homomorphisms to realized skeletons ensures that H_ℓ factors through $\mathsf{hull}_{\mathbb{A}'}$.

$\mathtt{Preceq}(\mathtt{m},\mathtt{n})$**:** Since $\mathbb{B}, \tau \models \phi_{\ell+1}$, $\tau(\mathtt{m}) \preceq_{\mathbb{B}} \tau(\mathtt{n})$, so H_ℓ is also a homomorphism from $\mathbb{A}_{\ell+1}$ to $\mathbb{B}$.

$\mathtt{Col}(\mathtt{m},\mathtt{n})$**:** In the non-failing case, ζ_ℓ maps $\sigma_\ell(\mathtt{m})$ and $\sigma_\ell(\mathtt{n})$ to nodes on the same strand. Thus, H_ℓ factors through the homomorphism from $\mathbb{A}_\ell$ to $\mathbb{A}_{\ell+1}$.

□

6 Security Goals

We turn now to our second result, which puts the pieces together.

Theorem 6.1 *Suppose that $G = \forall \vec{x}.\,(\phi \supset \exists \vec{y}.\,\psi)$ is a security goal in $\mathscr{L}(\Pi)$ where ϕ is satisfiable.*
Π achieves G iff, whenever $H\colon \mathsf{cs}(\phi) \mapsto \mathbb{B}$ is a shape, there is a σ such that $\mathbb{B}, \sigma \models \psi$.

Proof: 1. Suppose that Π achieves G. By Thm. 5.2, $\mathsf{cs}(\phi)$ is well-defined. By Lemma 4.3, $H\colon \mathsf{cs}(\phi) \mapsto \mathbb{B}$ implies that $\mathbb{B}$ satisfies ϕ. If $\mathbb{B}$ is a shape, it is realized. Since Π achieves G, $\mathbb{B}$ satisfies ψ.

2. Suppose that Π does not achieves G, so that there is a realized $\mathbb{C}$ which satisfies $\neg G$. Let $\psi = \bigvee_{1\le i\le \ell} \phi_i$. Using the Tarski satisfaction clauses and the disjointness of $\vec{x},\vec{y}$, we obtain a $\sigma_{\mathbb{C}}$ such that $\mathbb{C}, \sigma_{\mathbb{C}} \models \phi \wedge \bigwedge_{1\le i\le \ell} \neg\phi_i$.

Since $\mathbb{C}, \sigma_{\mathbb{C}} \models \phi$, there is a J such that $J\colon \mathsf{cs}(\phi) \mapsto \mathbb{C}$, and $\sigma_{\mathbb{C}} \restriction \mathsf{fv}(\phi) = J \circ \sigma_*$. Using Prop. 8 of the extended version of [8], $J = K \circ H$ where $H\colon \mathsf{cs}(\phi) \mapsto \mathbb{B}$ is a shape.

If this $\mathbb{B}$ satisfies any ϕ_i, then so would $\mathbb{C}$ by Lemma 4.3. □

Conclusion. We have explained a way to ensure that a protocol achieves a security goal G. We use the antecedent ϕ to choose a skeleton, namely $\mathsf{cs}(\phi)$. We then obtain the shapes accessible from $\mathsf{cs}(\phi)$, e.g. by using CPSA. If any shape does not satisfy any disjunct of the conclusion of G, then we have a counterexample. If no counterexample is found, then G is achieved.

In future work, we will apply this method to protocol transformation. It suggests a criterion to ensure that the result Π_2 of a protocol transformation preserves all goals achieved by its source protocol Π_1.

Acknowledgments. I am grateful to my colleagues, Leonard Monk, John Ramsdell, and Javier Thayer, for many relevant discussions. Marco Carbone gave valuable comments. John Ramsdell is the author of the CPSA implementation.

References

[1] B. Blanchet (2004): *Automatic proof of strong secrecy for security protocols*. In: *Security and Privacy, 2004. Proceedings. 2004 IEEE Symposium on*. IEEE CS Press, pp. 86–100.

[2] Bruno Blanchet (2008): *Vérification automatique de protocoles cryptographiques : modèle formel et modèle calculatoire. Automatic verification of security protocols: formal model and computational model*. Mémoire d'habilitation à diriger des recherches, Université Paris-Dauphine.

[3] I. Cervesato, N. A. Durgin, P. D. Lincoln, J. C. Mitchell & A. Scedrov (2000): *Relating Strands and Multiset Rewriting for Security Protocol Analysis*. In: *Proceedings, 13th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press.

[4] Michel Chein & Marie-Laure Mugnier (1992): *Conceptual Graphs: fundamental notions*. *Revue d'Intelligence Artificielle* 6, pp. 365–406.

[5] Michel Chein & Marie-Laure Mugnier (2004): *Concept Types and Coreference in Simple Conceptual Graphs*. In: *Conceptual Structures at Work: 12th International Conference on Conceptual Structures, ICCS 2004, Huntsville, AL, USA, July 19-23, 2004. Proceedings*, number 3127 in LNCS. pp. 303–318.

[6] Anupam Datta, Ante Derek, John C. Mitchell & Dusko Pavlovic (2005): *A Derivation System and Compositional Logic for Security Protocols*. *Journal of Computer Security* 13(3), pp. 423–482.

[7] Shaddin F. Doghmi, Joshua D. Guttman & F. Javier Thayer (2007): *Completeness of the Authentication Tests*. In: J. Biskup & J. Lopez, editors: *European Symposium on Research in Computer Security (ESORICS)*, number 4734 in LNCS. Springer-Verlag, pp. 106–121.

[8] Shaddin F. Doghmi, Joshua D. Guttman & F. Javier Thayer (2007): *Searching for Shapes in Cryptographic Protocols*. In: *Tools and Algorithms for Construction and Analysis of Systems (TACAS)*, number 4424 in LNCS. Springer, pp. 523–538. Extended version at URL:http://eprint.iacr.org/2006/435.

[9] Nancy Durgin, Patrick Lincoln, John Mitchell & Andre Scedrov (2004): *Multiset Rewriting and the Complexity of Bounded Security Protocols*. *Journal of Computer Security* 12(2), pp. 247–311. Initial version appeared in *Workshop on Formal Methods and Security Protocols*, 1999.

[10] Nancy Durgin, Patrick Lincoln, John Mitchell & Andre Scedrov (2004): *Multiset Rewriting and the Complexity of Bounded Security Protocols*. *Journal of Computer Security* 12(2), pp. 247–311. Initial version appeared in *Workshop on Formal Methods and Security Protocols*, 1999.

[11] Marcelo Fiore & Martín Abadi (2001): *Computing Symbolic Models for Verifying Cryptographic Protocols*. In: *Computer Security Foundations Workshop*.

[12] Sibylle Fröschle (2008): *Adding Branching to the Strand Space Model*. In: *Proceedings of EXPRESS'08*, ENTCS. Elsevier.

[13] Joshua D. Guttman (2001): *Security Goals: Packet Trajectories and Strand Spaces*. In: Roberto Gorrieri & Riccardo Focardi, editors: *Foundations of Security Analysis and Design*, *LNCS* 2171. Springer Verlag, pp. 197–261.

[14] Joshua D. Guttman (2009): *Cryptographic Protocol Composition via the Authentication Tests*. In: Luca de Alfaro, editor: *Foundations of Software Science and Computation Structures (FOSSACS)*, number 5504 in LNCS. Springer, pp. 303–317.

[15] Joshua D. Guttman (2009): *Transformations between Cryptographic Protocols*. In: P. Degano & L. Viganò, editors: *Automated Reasoning in Security Protocol Analysis, and Workshop on Issues in the Theory of Security (ARSPA-WITS)*, number 5511 in LNCS. Springer, pp. 107–123.

[16] Joshua D. Guttman, Jonathan C. Herzog, John D. Ramsdell & Brian T. Sniffen (2005): *Programming Cryptographic Protocols*. In: Rocco De Nicola & Davide Sangiorgi, editors: *Trust in Global Computing*, number 3705 in LNCS. Springer, pp. 116–145.

[17] Gavin Lowe (1997): *A Hierarchy of Authentication Specifications*. In: *10th Computer Security Foundations Workshop Proceedings*. IEEE Computer Society Press, pp. 31–43.

[18] Jonathan K. Millen & Vitaly Shmatikov (2001): *Constraint Solving for Bounded-Process Cryptographic Protocol Analysis*. In: *8th ACM Conference on Computer and Communications Security (CCS '01)*. ACM, pp. 166–175.

[19] Lawrence C. Paulson (1998): *The Inductive Approach to Verifying Cryptographic Protocols*. *Journal of Computer Security* Also Report 443, Cambridge University Computer Lab.

[20] Michaël Rusinowitch & Mathieu Turuani (2001): *Protocol Insecurity with Finite Number of Sessions is NP-*

Complete. In: *Computer Security Foundations Workshop*. pp. 174–. Available at http://csdl.computer.org/comp/proceedings/csfw/2001/1146/00/11460174abs.htm.

[21] Peter Selinger (2001): *Models for an adversary-centric protocol logic*. *Electr. Notes Theor. Comput. Sci.* 55(1). Available at http://www.elsevier.com/gej-ng/31/29/23/83/27/show/Products/notes/index.htt#007.

A Messages and Protocols

Messages are abstract syntax trees in the usual way:

1. Let ℓ and r be the partial functions such that for $t = \{|t_1|\}_{t_2}$ or $t = tag\ t_1\hat{}t_2$, $\ell(t) = t_1$ and $r(t) = t_2$; and for $t \in \mathfrak{A}_0$, ℓ and r are undefined.
2. A *path* p is a sequence in $\{\ell, r\}^*$. We regard p as a partial function, where $\langle\rangle = \mathsf{Id}$ and $\mathsf{cons}(f, p) = p \circ f$. When the rhs is defined, we have: 1. $\langle\rangle(t) = t$; 2. $\mathsf{cons}(\ell, p)(t) = p(\ell(t))$; and 3. $\mathsf{cons}(r, p)(t) = p(r(t))$.
3. p *traverses a key edge* in t if $p_1(t)$ is an encryption, where $p = p_1{}^\frown\langle r\rangle{}^\frown p_2$.
4. t_0 *is an ingredient of* t, written $t_0 \sqsubseteq t$, if $t_0 = p(t)$ for some p that does not traverse a key edge in t.
5. t_0 *appears in* t, written $t_0 \ll t$, if $t_0 = p(t)$ for some p.

A message t_0 *originates* at a node n_1 if (1) n_1 is a transmission node; (2) $t_0 \sqsubseteq \mathsf{msg}(n_1)$; and (3) whenever $n_0 \Rightarrow^+ n_1$, $t_0 \not\sqsubseteq \mathsf{msg}(n_0)$.

In the tree model of messages, to apply a homomorphism, we walk through, copying the tree, but inserting $\alpha(a)$ every time an atom a is encountered, and inserting $\alpha(x)$ every time that an indeterminate x is encountered.

Protocols. A *protocol* Π is a finite set of strands which includes $\mathsf{Lsn}[k]$, representing the roles of the protocol.

A principal executing a role such as the initiator's role in Fig. 1 may be partway through its run; for instance, it may have executed the first transmission node without "yet" having executed its second event, the reception node.

Definition A.1 *Node n is a* role node *of Π if n lies on some $\rho \in \Pi$.*

Let n_j be a role node of Π of the form $n_1 \Rightarrow \ldots \Rightarrow n_j \Rightarrow \ldots$. Node m_j is an instance *of n_j if, for some homomorphism α, the strand of m_j, up to m_j, takes the form: $\alpha(n_1) \Rightarrow \ldots \Rightarrow \alpha(n_j) = m_j$.*

That is, messages and their directions—transmission or reception—must agree up to node j. However, any remainders of the two strands beyond node j are unconstrained. They need not be compatible. When a protocol allows a principals to decide between different behaviors after step j, based on the message contents of their run, then this definition represents branching [12, 16]. At step j, one doesn't yet know which branch will be taken.