Skeletons, Homomorphisms, and Shapes: Characterizing Protocol Executions [1]

Shaddin F. Doghmi[2,5]

Stanford University
Stanford, CA, USA

Joshua D. Guttman[3]

The MITRE Corporation
Bedford, MA, USA

F. Javier Thayer[4]

The MITRE Corporation
Bedford, MA, USA

Abstract

In this paper we develop a framework, based on strand spaces, for reasoning about cryptographic protocols and characterizing their executions. We define skeletons, homomorphisms, and shapes. Skeletons model partial information about regular (honest) behavior in an execution of a cryptographic protocol. A homomorphism between skeletons is an information-preserving map. Much protocol analysis may be regarded as an exploration of the properties of the category of skeletons and homomorphisms. A set of skeletons can characterize all runs of the protocol; the smallest such set is the set of shapes. This approach is a foundation for mechanizing protocol analysis.

Keywords: protocol analysis, strand spaces, enumerating protocol executions, mechanizing protocol analysis.

1 Introduction

Most protocol analysis tools and techniques operate on formulas, proving or refuting formulas that express the security goals of a protocol in a specific logic. Starting from some initial assumptions, theorem proving or model checking (such as in [9])

[1] Supported by the National Security Agency and by MITRE-Sponsored Research.

[2] Email: shaddin@cs.stanford.edu

[3] Email: guttman@mitre.org

[4] Email: jt@mitre.org

[5] This research was conducted while the author was an employee at The MITRE Corporation.

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

techniques can be used to check if a certain security property follows from a protocol definition. In this paper, we take a different approach to this problem.

Instead of checking each security property individually, we characterize all protocol executions compatible with the initial assumptions. The resulting *characterization* is a set of protocol runs that is representative of all possible protocol runs. These representatives are the minimal, essentially different executions of the protocol. Some advantages of this approach are:

- We find that the proofs for different security properties often duplicate the same work. With this approach, all the work needed to characterize possible runs can be done once. Then it is easy for a human analyst or simple tool to "read off" the value of any security predicate from the characterization by evaluating that predicate on each run in the characterization.
- Security properties of the protocol that were not anticipated by the designer will become apparent.
- The analyst interested in a certain security property can see all the possible attacks/counterexamples as opposed to a single attack.
- Since all possible protocol runs are represented, this gives the protocol designer more insight into the effect of the different primitives used to construct the protocol.

In this paper, we will present a framework, based on strand spaces, for analyzing protocols and characterizing their executions. While a generalized notion of our *characterizations* can capture both authentication and secrecy properties, we focus here on a simpler notion that captures authentication properties exclusively. We will discuss the applicability to secrecy elsewhere. In practice, any algorithm for constructing characterizations must reason about both secrecy and authentication.

As motivation for this framework, consider a protocol analyst presented with some initial assumptions about a protocol run. Often this is the experience of a single participant and some secrecy and freshness assumptions. The analyst can then repeatedly apply inference rules such as the authentication tests [4] in order to infer more about the structure of the protocol run.

At any point in the analysis, the analyst possesses partial information about the structure of possible protocol runs. We will represent this partial information in structures we will call *skeletons*. We will also define a notion of *homomorphism* between skeletons; homomorphisms are information-preserving maps. Thus, much of protocol analysis can be expressed in terms of skeletons and homomorphisms between them.

We discuss the actual algorithms used to construct these characterizations in [2]. In the present paper, we define the underlying framework for the algorithms. Given a set of initial assumptions (an initial skeleton), we will define how a set of protocol runs can *characterize* all possible runs. Furthermore, we will show that there is a minimum such *characterization*: the set of *shapes*. They are the minimal, essentially different executions possible for the protocol.

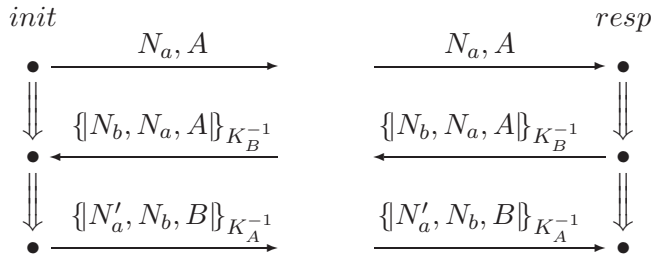

Fig. 1. ISO Candidate Protocol

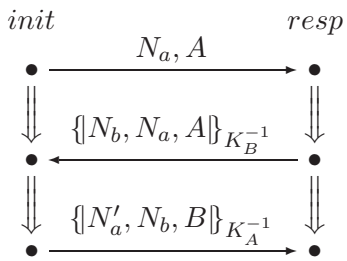

Fig. 2. Bundle: Intended Run

2 A Motivating Example

The protocol shown in Figure 1 was a candidate considered by an ISO committee as a *pure* authentication protocol [3]. No shared secret is achieved. It uses asymmetric cryptography for digital signatures, and is intended to assure each principal that the expected partner intends to engage in a communication session. Figure 1 gives the roles of the protocol. Each of the two vertical columns describes a behavior that principals engaging in the protocol may follow. The column on the left originates a session, while the column on the right is the responder's behavior. Any principal Alice, Bob, etc., may engage in either of these roles, or possibly in several sessions of each role at the same time. An actual run of the protocol consists of any finite number of instances of these roles, connected by message transmissions and (possibly) actions of an adversary.

We call a vertical column connected by double arrows $\bullet \Rightarrow \bullet$ a *strand*, and use it to represent a single local session of a principal executing a protocol, or alternatively a sequence of adversary activities.

One kind of execution simply connects the corresponding arrows, and this models a session in which the messages sent by the principals, called A and B here, reach their destinations (Figure 2). We call such an execution a *bundle*. This is a small bundle. There are infinitely many larger bundles consisting of multiple copies of this diagram, possibly with different values of the parameters. However, the larger executions are only inessentially different from Figure 2.

However, this protocol was rejected by the committee, because they found that another, essentially different execution is possible [3], shown in Figure 3. Since it was discovered by the Canadian representatives to the committee, it is sometimes called

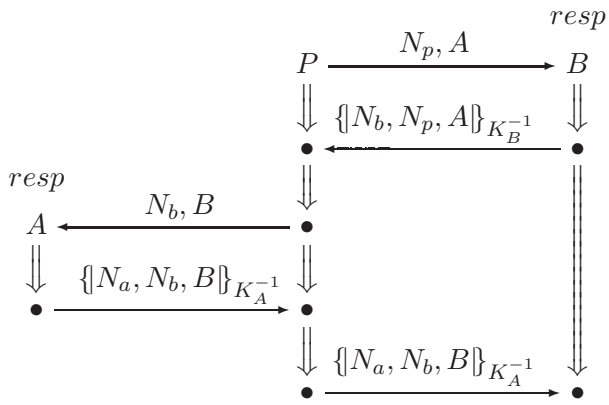

Fig. 3. Bundle: The Canadian Attack

the Canadian attack [3]. The attacker is denoted by P. The attack involves two runs of the responder role, both stimulated by messages delivered by the adversary.

This attack constitutes a failure of authentication from the perspective of responder B. Even if both B and A had uncompromised private keys, the protocol does not guarantee to the responder B that A is running a corresponding instance of the initiator role. In this attack, A is running an instance of the responder role, although only the first two steps of the role have occurred; A is waiting for the third message for this session, though it will never be delivered.

Suppose that these are *all* of the minimal, essentially different executions possible containing a responder run using uncompromised private keys. In that case, we can read off the degree of authentication that B obtains. B knows at the end of a responder session that there has been an A session that agrees with B's session on the participant's identities and the parameters N_a, N_b. However, B cannot know whether it is an initiator role of length three or a responder role of length two, since executions exist with each. Since the protocol designers wanted to ensure that B would know that there is a matching initiator run, the protocol was too weak to meet the goal.

3 The Idea of Shape

In the ensuing discussion we will use the term *regular strand* to refer to a run of some role of the protocol. Likewise we will use *regular node* to refer to a send or receive event occuring on a regular strand. We also use *regular behavior* to refer to all regular nodes in a particular run of the protocol as well as their causal ordering. We will also use the notation term(n) to refer to the message sent or received on node n.

In practice, protocols have remarkably few shapes. The ISO-reject protocol introduced above has two if we take the view of a responder, asking what global behaviour must have occured if B has had a local run of the protocol.

In the ISO reject protocol, let us suppose that B's nonce N_b has been freshly chosen and that the the party that B wishes to respond to, A, has an uncompromized private key K_A^{-1}. Assume that the responder B completes a run of the protocol,

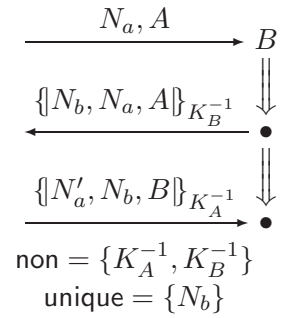

Fig. 4. ISO Candidate Responder Skeleton

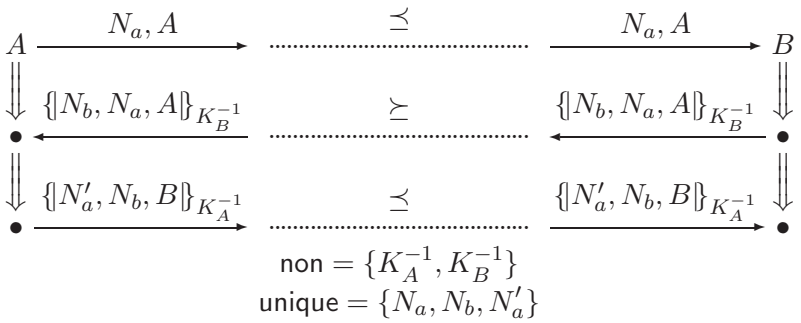

Fig. 5. ISO Candidate Intended Shape

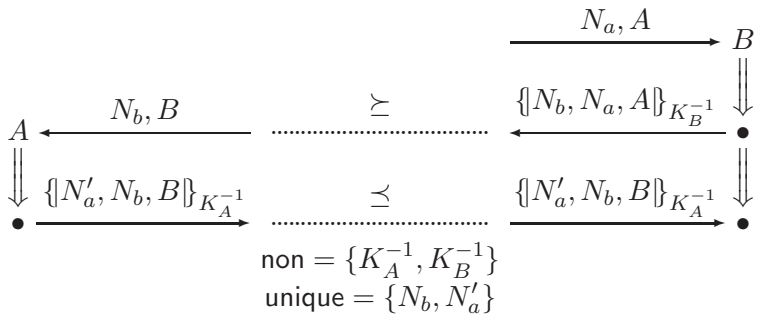

Fig. 6. ISO Candidate Attack Shape

executing the strand shown in 4. What must have happened elsewhere in the network? There are two possibilities.

3.0.1 ISO Shapes

The last message received by the responder has been signed by an uncompromised private key K_A^{-1}. Let us consider the case where that message originated on an initiator strand. In this case, the initiator A must have had a partially matching strand, with the messages sent and received in the expected order. This is depicted in Fig. 5. The order of message sends and received can be deduced via the freshness assumption on nonce N_b, as well an assumption that any regular strand generates

fresh nonces, in this case N_a and N'_a. The order is depicted by the arrows of both kinds and the connecting symbols $\preceq$. These symbols mean that the endpoints are ordered, but that other behavior may intervene, whether adversary strands or regular strands.

Another possibility is that the signed message originated on a responder strand. In this case, we can use similar analysis to deduce that the events in Fig. 6 must have occured, in that order.

There is no alternative to these two shapes. Any diagram containing the responder strand depicted in Fig. 4 must contain either an intiator strand with the events ordered as shown in Fig. 5, or a responder strand with the events ordered as in Fig. 6.

Such a diagram is a *shape*. A shape consists of the regular strands of some execution, forming a *minimal* set containing the initial regular strands (in this case, just the right-hand column). Possible executions may freely add adversary behavior. Each shape is relative to assumptions about keys and freshness, in this case that K_A^{-1} is uncompromised and N_b freshly chosen, and furthermore that regular strands generate fresh nonces.

In searching for shapes, one starts from some initial set of strands. Typically, the initial set is a singleton, which we refer to as the "point of view" of the analysis.

Skeletons, Homomorphisms, Shapes.
Newly introduced terminology is in **boldface** in this section.

A **skeleton** $\mathbb{A}$ is (1) a finite set of regular nodes, equipped with additional information. The additional information consists of (2) a partial order $\preceq_{\mathbb{A}}$ on the nodes indicating causal precedence; (3) a set of keys $\mathsf{non}_{\mathbb{A}}$; and (4) a set of atomic values $\mathsf{unique}_{\mathbb{A}}$. Values in $\mathsf{non}_{\mathbb{A}}$ must originate nowhere in $\mathbb{A}$, whereas those in $\mathsf{unique}_{\mathbb{A}}$ originate at most once in $\mathbb{A}$.[6] (See Def. 5.1.)

We say a skeleton $\mathbb{A}$ is **realized** if it has precisely the regular behavior of some execution. Every message received by a regular strand either should have been sent previously, or should be constructable by the adversary using messages sent previously.

Example 3.1 Fig. 5 shows skeleton $\mathbb{A}_{iso1}$, with $\mathsf{non}_{\mathbb{A}_{iso1}} = \{K_A^{-1}, K_B^{-1}\}$ and $\mathsf{unique}_{\mathbb{A}_{iso1}} = \{N_a, N_b, N'_a\}$. $\mathbb{A}_{iso1}$ is a realized skeleton.

Fig. 6 shows skeleton $\mathbb{A}_{iso2}$. This skeleton is realized, and depicts regular behavior in the attack on the protocol.

The single responder strand depicted in Fig. 4, also forms a skeleton $\mathbb{A}_b$ with $\mathsf{non}, \mathsf{unique}$ as shown. $\mathbb{A}_b$ is not realized.

The first two nodes of $\mathbb{A}_b$ also form a skeleton $\mathbb{A}_{b_2}$. This skeleton is realized, as the adversary can prepare the incoming message of its first node, and discard the outgoing message of its second node.

A **homomorphism** is a map H from $\mathbb{A}_0$ to $\mathbb{A}_1$, written $H\colon \mathbb{A}_0 \to \mathbb{A}_1$. We represent it as a pair of maps (ϕ, α), where ϕ maps the nodes of $\mathbb{A}_0$ into those of $\mathbb{A}_1$, and α is a **replacement** mapping atomic values into atomic values. We write

[6] When $n \Rightarrow^* n'$ and $n' \in \mathbb{A}$, we require $n \in \mathbb{A}$ and $n \preceq_{\mathbb{A}} n'$.

$t \cdot \alpha$ for the result of applying a replacement α to a message t. $H = (\phi, \alpha)$ is a homomorphism iff: (1) ϕ respects strand structure, and $\mathrm{term}(n) \cdot \alpha = \mathrm{term}(\phi(n))$ for all $n \in \mathbb{A}_0$; (2) $m \preceq_{\mathbb{A}_0} n$ implies $\phi(m) \preceq_{\mathbb{A}_1} \phi(n)$; (3) $\mathsf{non}_{\mathbb{A}_0} \cdot \alpha \subseteq \mathsf{non}_{\mathbb{A}_1}$; and (4) $\mathsf{unique}_{\mathbb{A}_0} \cdot \alpha \subseteq \mathsf{unique}_{\mathbb{A}_1}$. (Defs. 4.1, 5.6.)

Homomorphisms are *information-preserving* transformations. Each skeleton $\mathbb{A}_0$ describes the realized skeletons reachable from $\mathbb{A}_0$ by homomorphisms. Since homomorphisms compose, if $H\colon \mathbb{A}_0 \to \mathbb{A}_1$ then any realized skeleton accessible from $\mathbb{A}_1$ is accessible from $\mathbb{A}_0$. Thus, $\mathbb{A}_1$ preserves the information in $\mathbb{A}_0$: $\mathbb{A}_1$ describes a subset of the realized skeletons described by $\mathbb{A}_0$.

A homomorphism may supplement the strands of $\mathbb{A}_0$ with additional behavior in $\mathbb{A}_1$; it may affect atomic parameter values; and it may identify different nodes together, if their strands are compatible in messages sent and positions in the partial ordering.

Example 3.2 The map $H\colon \mathbb{A}_b \to \mathbb{A}_{iso1}$ embedding the skeleton in Fig. 4 into the skeleton in Fig. 5 is a homomorphism. Likewise if we embed the first two nodes of B's strand (rather than all of $\mathbb{A}_b$) into $\mathbb{A}_{iso1}$.

Consider the variant $\mathbb{A}'$ of skeleton $\mathbb{A}_{iso1}$ that we would get by renaming the variable A to B (that is, we are identifying A and B). There is a homomorphism H' mapping $\mathbb{A}_b$ to $\mathbb{A}'$, embedding as did H, but also renaming A to B.

A homomorphism $H = (\phi, \alpha)$ is **nodewise injective** if the function ϕ on nodes is injective. The nodewise injective homomorphisms determine a useful partial order on homomorphisms: When for some nodewise injective H_1, $H_1 \circ H = H'$, we write $H \leq_n H'$. If $H \leq_n H' \leq_n H$, then H and H' are isomorphic. We say a homomorphism H *characterizes* all realized homomorphisms H' such that $H \leq_n H'$.

A homomorphism $H\colon \mathbb{A}_0 \to \mathbb{A}_1$ is a **shape** iff (a) $\mathbb{A}_1$ is realized and (b) H is $\leq_n$-minimal among homomorphisms from $\mathbb{A}_0$ to realized skeletons. If H is a shape, and we can factor H into $\mathbb{A}_0 \stackrel{H_0}{\to} \mathbb{A}' \stackrel{H_1}{\to} \mathbb{A}_1$, where $\mathbb{A}'$ is realized, then $\mathbb{A}'$ cannot contain fewer nodes than $\mathbb{A}_1$, or identify fewer atomic values. $\mathbb{A}_1$ is as small and as general as possible. (Def. 7.8.)

We call a *skeleton* $\mathbb{A}_1$ a shape when the homomorphism H (usually an embedding) is understood. In this looser sense, both Fig. 5 and Fig. 6 depict shapes. Strictly, the embeddings $H_{iso1}\colon \mathbb{A}_b \to \mathbb{A}_{iso1}$ and $H_{iso2}\colon \mathbb{A}_b \to \mathbb{A}_{iso2}$ are the shapes.

Shapes exist below realized skeletons: If $H\colon \mathbb{A}_0 \to \mathbb{A}_1$ with $\mathbb{A}_1$ realized, then the set of shapes H_1 with $H_1 \leq_n H$ is finite and non-empty.

4 Background

In this section we define the core strand space notions.

We define our algebra A of terms. Terms in A are built from atomic terms via constructors. The atomic terms are partitioned into the types *principals*, *texts*, *keys*, and *nonces*. An inverse operator is defined on keys. There may be additional operations on atoms, such as an injective *public key of* function or an injective *long term shared key of* function mapping principals to keys. Atoms serve as indeterminates (variables), and are written in italics (e.g. a, N_a, K^{-1}). We assume A contains infinitely many atoms of each type.

Terms in A are freely built from atoms using *concatenation* and *encryption*. The concatenation of t_0 and t_1 is written t_0, t_1. Encryption takes a term t and an atomic key K, and yields a term written as $\{\!|t|\!\}_K$. Instead of general substitutions, we use *Replacements* for simplicity. *Replacements* have only atoms in their range:

Definition 4.1 [Replacement, Application] A *replacement* is a function α mapping atoms in A to atoms in A, such that (1) for every atom a, $\alpha(a)$ is an atom of the same type as a, and (2) α is a homomorphism with respect to the operations on atoms, e.g. in the case of inverse keys, for every key K, $K^{-1} \cdot \alpha = (K \cdot \alpha)^{-1}$.

The *application* of α to t, written $t \cdot \alpha$, homomorphically extends α's action on atoms. More explicitly, if $t = a$ is an atom, then $a \cdot \alpha = \alpha(a)$; and:

$$(t_0, t_1) \cdot \alpha = (t_0 \cdot \alpha), (t_1 \cdot \alpha)$$
$$(\{\!|t|\!\}_K) \cdot \alpha = \{\!|t \cdot \alpha|\!\}_{K \cdot \alpha}$$

Application distributes through pairing and sets. Thus, $(x, y) \cdot \alpha = (x \cdot \alpha, y \cdot \alpha)$, and $S \cdot \alpha = \{x \cdot \alpha : x \in S\}$. If $x \notin \mathsf{A}$ is a simple value such as an integer or a symbol, then $x \cdot \alpha = x$.

For the purposes of this paper and upcoming work, restricting ourselves to replacements is a convenient simplification. However, our model is extensible to using a more general notion of substitution. We may concern ourselves with that in future work.

Since replacements map atoms to atoms, not to compound terms, unification is very simple. A replacement α unifies t_1 and t_2 if $t_1 \cdot \alpha = t_2 \cdot \alpha$. Two terms are unifiable if and only if they have the same abstract syntax tree structure, with the same type for atoms at corresponding leaves. If t_1, t_2 are unifiable, then $t_1 \cdot \alpha$ and $t_2 \cdot \beta$ are unifiable. Any two unifiable terms t_1, t_2 have a unique (up to renaming) most general unifier.

The direction $+$ means transmission, and the direction $-$ means reception:

Definition 4.2 [Strand Spaces] A *direction* is one of the symbols $+, -$. A *directed term* is a pair (d, t) with $t \in \mathsf{A}$ and d a direction, normally written $+t, -t$. $(\pm\mathsf{A})^*$ is the set of finite sequences of directed terms.

A *strand space* over A is a structure containing a set Σ and two mappings: a trace mapping $\mathsf{tr} : \Sigma \to (\pm\mathsf{A})^*$ and a replacement application operator $(s, \alpha) \to s \cdot \alpha$ such that (1) $\mathsf{tr}(s \cdot \alpha) = (\mathsf{tr}(s)) \cdot \alpha$, and (2) $s \cdot \alpha = s' \cdot \alpha$ implies $s = s'$. Elements of Σ are called *strands*.

By condition (2), any non-trivial Σ has infinitely many copies of each strand s, i.e. strands s' with $\mathsf{tr}(s') = \mathsf{tr}(s)$. Note that in this paper, when we refer to a strand we are implicitly also referring to its trace.

Definition 4.3 A *penetrator strand* has a trace of one of the following forms:

M: $\langle +t \rangle$ where $t \in$text, principal,nonce	K: $\langle +K \rangle$
C: $\langle -g,\ -h,\ +g, h \rangle$	S: $\langle -g, h,\ +g,\ +h \rangle$
E: $\langle -K,\ -h,\ +\{\!|h|\!\}_K \rangle$	D: $\langle -K^{-1},\ -\{\!|h|\!\}_K,\ +h \rangle$.

If s is a penetrator strand, then $s \cdot \alpha$ is a penetrator strand of the same kind.

Definition 4.4 [Protocols] A *protocol* $\Pi = \langle R, n, u \rangle$ consists of (1) a finite set R of strands called the *roles* of the protocol, and (2) for each role $r \in R$, two sets of atoms n_r, u_r giving *origination data* for r. The *regular strands* Σ_Π *over* Π consists of all strands with traces $r \cdot \alpha$ for some $r \in R$ and some replacement α.

Example 4.5 Figure 1 depicts the ISO Reject protocol. Note that the origination data for the initiator role indicates that any honest initiator has an uncompromised private key K_A^{-1}, and that he generates fresh nonces N_a and N'_a. Similarly any honest responder has an uncompromised private key K_B^{-1} and generates a fresh nonce N_b.

A *node* is a pair $n = (s, i)$ where $i \leq \mathsf{length}(\mathsf{tr}(s))$; $\mathsf{strand}(s, i) = s$; and the *direction* and *term* of n are those of $\mathsf{tr}(s)(i)$. We call a node $n = (s, i)$ *regular* if s is a regular strand and the relevant protocol is understood. We prefer to write $s \downarrow i$ for the node $n = (s, i)$. The set $\mathcal{N}$ of all nodes forms a directed graph $\mathcal{G} = \langle \mathcal{N}, (\rightarrow \cup \Rightarrow) \rangle$ with edges $n_1 \rightarrow n_2$ for communication (with the same term, directed from positive to negative node) and $n_1 \Rightarrow n_2$ for succession on the same strand.

The *subterm* relation, written $\sqsubset$, is the least reflexive, transitive relation such that (1) $t_0 \sqsubset t_0, t_1$; (2) $t_1 \sqsubset t_0, t_1$; and (3) $t \sqsubset \{\!|t|\!\}_K$. Notice, however, $K \not\sqsubset \{\!|t|\!\}_K$ unless (anomalously) $K \sqsubset t$. We say an atom a occurs in term t if $a \sqsubset t$. We say that a key K is *used for encryption* in a term t if for some t_0, $\{\!|t_0|\!\}_K \sqsubset t$. If a occurs in t or is used in t, then a is *mentioned* in t.

A term t *originates* at node n if n is positive, $t \sqsubset \mathrm{term}(n)$, and $t \not\sqsubset \mathrm{term}(m)$ whenever $m \Rightarrow^+ n$. Thus, t originates on n if t is part of a message transmitted on n, and t was neither sent nor received previously on this strand.

Definition 4.6 [Bundle] A finite acyclic subgraph $\mathcal{B} = \langle \mathcal{N}_\mathcal{B}, (\rightarrow_\mathcal{B} \cup \Rightarrow_\mathcal{B}) \rangle$ of $\mathcal{G}$ is a *bundle* if (1) if $n_2 \in \mathcal{N}_\mathcal{B}$ is negative, then there is a unique $n_1 \in \mathcal{N}_\mathcal{B}$ with $n_1 \rightarrow_\mathcal{B} n_2$; and (2) if $n_2 \in \mathcal{N}_\mathcal{B}$ and $n_1 \Rightarrow n_2$, then $n_1 \Rightarrow_\mathcal{B} n_2$. When $\mathcal{B}$ is a bundle, $\preceq_\mathcal{B}$ is the reflexive, transitive closure of $(\rightarrow_\mathcal{B} \cup \Rightarrow_\mathcal{B})$.

A bundle $\mathcal{B}$ is *over* $\Pi = \langle R, n, u \rangle$ if for every $s \downarrow i \in \mathcal{B}$, (1) either $s \in \Sigma_\Pi$ or s is a penetrator strand; (2) if $s = r \cdot \alpha$ and $a \in n_r \cdot \alpha$, then a does not originate in $\mathcal{B}$; and (3) if $s = r \cdot \alpha$ and $a \in u_r \cdot \alpha$, then a originates at most once in $\mathcal{B}$.

Example 4.7 Figures 2 and 3 depict two different bundles for the protocol in Figure 1

Proposition 4.8 *Let $\mathcal{B}$ be a bundle. $\preceq_\mathcal{B}$ is a well-founded partial order. Every non-empty set of nodes of $\mathcal{B}$ has $\preceq_\mathcal{B}$-minimal members. If α is a replacement, then $\mathcal{B} \cdot \alpha$ is a bundle.*

We will also define the sub-bundle relation between bundles.

Definition 4.9 A bundle $\mathcal{B}_1$ is a sub-bundle of bundle $\mathcal{B}_2$ if $\mathcal{B}_1$ is a subgraph of $\mathcal{B}_2$, up to (injective) renaming of nodes.

5 Skeletons and Homomorphisms

In this section we will define the framework of skeletons and homomorphisms.

A preskeleton describes the regular (honest) parts of a set of bundles.

Definition 5.1 A four-tuple $\mathbb{A} = (\mathsf{node}, \preceq, \mathsf{non}, \mathsf{unique})$ is a *preskeleton* if:

(i) node is a finite set of regular nodes; $n_1 \in \mathsf{node}$ and $n_0 \Rightarrow^+ n_1$ implies $n_0 \in \mathsf{node}$;

(ii) $\preceq$ is a partial ordering on node such that $n_0 \Rightarrow^+ n_1$ implies $n_0 \preceq n_1$;

(iii) non is a set of keys where if $K \in \mathsf{non}$, then for all $n \in \mathsf{node}$, $K \not\sqsubset \mathrm{term}(n)$, and for some $n' \in \mathsf{node}$, either K or K^{-1} is used in $\mathrm{term}(n')$;

(iv) unique is a set of atoms where if $a \in \mathsf{unique}$, for some $n \in \mathsf{node}$, $a \sqsubset \mathrm{term}(n)$.

A preskeleton $\mathbb{A}$ is a *skeleton* if in addition:

(v) $a \in \mathsf{unique}$ implies a originates at no more than one $n \in \mathsf{node}$.

A skeleton is similar to the notion of semi-bundle in [9]. The reason we define both preskeletons and skeletons is because preskeletons are useful as precursors to skeletons. One can transform a preskeleton to a skeleton by unifying and combining strands that originate the same atom $a \in \mathsf{unique}$.

We select components of a preskeleton using subscripts. For instance, if $\mathbb{A} = (\mathsf{node}, R, S, S')$, then $\preceq_{\mathbb{A}}$ means R and $\mathsf{unique}_{\mathbb{A}}$ means S'. We write $n \in \mathbb{A}$ to mean $n \in \mathsf{node}_{\mathbb{A}}$, and we say that a strand s is in $\mathbb{A}$ when at least one node of s is in $\mathbb{A}$. The $\mathbb{A}$-height of s is the number of nodes of s in $\mathbb{A}$. By Clauses iii and iv, $\mathsf{unique}_{\mathbb{A}} \cap \mathsf{non}_{\mathbb{A}} = \emptyset$.

We will define the sub-[pre]skeleton relation analogously to the sub-bundle relation:

Definition 5.2 A [pre]skeleton $\mathbb{A}$ is a sub-[pre]skeleton of $\mathbb{A}'$ if each of $\mathsf{node}_{\mathbb{A}}, \preceq_{\mathbb{A}}, \mathsf{non}_{\mathbb{A}}, \mathsf{unique}_{\mathbb{A}}$ is a subset of $\mathsf{node}_{\mathbb{A}'}, \preceq_{\mathbb{A}'}, \mathsf{non}_{\mathbb{A}'}, \mathsf{unique}_{\mathbb{A}'}$ respectively, up to (injective) renaming of nodes.

Bundles correspond to certain skeletons:

Definition 5.3 Bundle $\mathcal{B}$ *realizes* skeleton $\mathbb{A}$ if (1) the nodes of $\mathbb{A}$ are precisely the regular nodes of $\mathcal{B}$; (2) $n \preceq_{\mathbb{A}} n'$ just in case $n, n' \in \mathsf{node}_{\mathbb{A}}$ and $n \preceq_{\mathcal{B}} n'$; (3) $K \in \mathsf{non}_{\mathbb{A}}$ just in case $K \not\sqsubset \mathrm{term}(n)$ for any $n \in \mathcal{B}$ but K or K^{-1} is used in some $n' \in \mathcal{B}$; (4) $a \in \mathsf{unique}_{\mathbb{A}}$ just in case a originates uniquely in $\mathcal{B}$. if some $\mathcal{B}$ realizes $\mathbb{A}$ we say that $\mathbb{A}$ is a *realized skeleton*.

Example 5.4 The intended run bundle in Figure 2 realizes the skeleton in Figure 5. Likewise, the attack bundle in Figure 3 realizes the skeleton in Figure 6.

In fact, a bundle completely determines the skeleton it realizes.

Proposition 5.5 *If $\mathcal{B}$ is a bundle, then there is a unique skeleton that it realizes. By condition (4), $\mathcal{B}$ does not realize $\mathbb{A}$ if $\mathbb{A}$ is a preskeleton but not a skeleton.*

Homomorphisms.

Since preskeletons represent partial-information about a protocol run, it would be useful to define information-preserving maps between them: *homomorphisms*

Definition 5.6 Let $\mathbb{A}_0, \mathbb{A}_1$ be preskeletons, α a replacement, $\phi\colon \mathsf{node}_{\mathbb{A}_0} \to \mathsf{node}_{\mathbb{A}_1}$. $H = [\phi, \alpha]$ is a *homomorphism* if

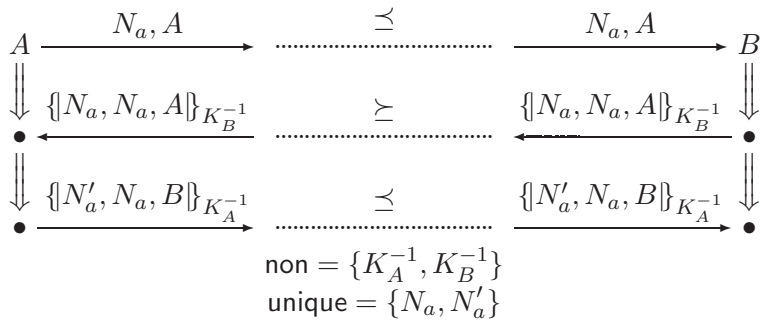

Fig. 7. Result of a degenerate homomorphism

(i) For all $n \in \mathbb{A}_0$, $\mathrm{term}(\phi(n)) = \mathrm{term}(n) \cdot \alpha$;

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

(iii) $n \preceq_{\mathbb{A}_0} m$ implies $\phi(n) \preceq_{\mathbb{A}_1} \phi(m)$;

(iv) $\mathsf{non}_{\mathbb{A}_0} \cdot \alpha \subset \mathsf{non}_{\mathbb{A}_1}$;

(v) $\mathsf{unique}_{\mathbb{A}_0} \cdot \alpha \subset \mathsf{unique}_{\mathbb{A}_1}$.

We write $H\colon \mathbb{A}_0 \to \mathbb{A}_1$ when H is a homomorphism from $\mathbb{A}_0$ to $\mathbb{A}_1$.

Homomorphisms can transform preskeletons to skeletons by contracting nodes that originate the same $a \in \mathsf{unique}_{\mathbb{A}}$. Furthermore, homomorphisms can transform preskeletons to realized skeletons.

There are many runs of the protocol compatible with a set of starting assumptions represented by a preskeleton $\mathbb{A}$. Homomorphisms from $\mathbb{A}$ to realized skeletons describe how $\mathbb{A}$ is part of a protocol run.

Definition 5.7 We say a homomorphism H realizes preskeleton $\mathbb{A}$ if it maps $\mathbb{A}$ to a realized skeleton $\mathbb{A}'$. In this case, we say $\mathbb{A}$ is *realizable*. Realized skeletons are preskeletons that are realizable via the identity homomorphism.

6 Degeneracy

We use the notion of unique-origination to represent fresh generation of values such as nonces and session keys. This necessitates that we restrict the set of homomorphisms we are interested in to homomorphisms that respect this intended real-world meaning of unique-origination.

Definition 6.1 [Degenerate Homomorphism] A replacement α is *degenerate for* $\mathbb{A}$ if there are distinct atoms a, b and a strand s where (1) $a \in \mathsf{unique}_{\mathbb{A}}$ originates at $s \downarrow i$ in $\mathbb{A}$, (2) b occurs on $s \downarrow j$ for $j \leq i$, and (3) $a \cdot \alpha = b \cdot \alpha$.

$H = [\phi, \alpha]\colon \mathbb{A}_0 \to \mathbb{A}$ is *degenerate* if α is degenerate for $\mathbb{A}_0$.

A degenerate replacement identifies a uniquely originating atom with some other atom already known at the time it is chosen. For example, a homomorphism that maps the skeleton in Figure 5 to the skeleton in Figure 7 by identifying N_a and N_b is degenerate.

Degenerate homomorphisms are of negligible probability relative to stochastic models for protocols [5]. In the rest of this paper, the reader can assume every reference to homomorphisms is in fact restricted to non-degenerate homomorphisms.

In the same spirit, we are only interested in [pre]skeletons/bundles that respect the real-world meaning of unique-origination.

Definition 6.2 [Degenerate Preskeleton/Bundle] A [pre]skeleton/bundle is degenerate if it contains a strand $r \cdot \alpha$ of role r such that (1) $a \in u_r$ originates at $r \downarrow i$ (2) b occurs on $r \downarrow j$ for $j \leq i$, and (3) $a \cdot \alpha = b \cdot \alpha$.

The skeleton in Figure 7 is degenerate. Likewise, the reader can assume that every mention of [pre]skeletons/bundles is restricted to non-degenerate [pre]skeletons/bundles.

7 Shapes

7.1 Needham Schroeder

We saw that the ISO reject protocol has two shapes, one corresponding to the intended run and another corresponding to the attack. The Needham-Schoeder-Lowe [7,6] protocol has only one shape, corresponding to the intended run. This holds for NSL whether we take the point of view of a responder B, asking what global behavior must have occurred if B has had a local run of the protocol, or whether we start from a local run of an originator A. In either case, the other party must have had a matching run. A, however, can never be sure that the last message it sends was received by B, as A is no longer expecting to receive any further messages. Uniqueness of shape is perhaps not surprising for as strong a protocol as Needham-Schroeder-Lowe.

However, even a flawed protocol such as the original Needham-Schroeder protocol, depicted in Figure 8 may have a unique shape, shown in Fig. 9, despite both the existence of an intended run depicted in Fig. 10 and an attack depicted in Fig. 11. Here we are studying Needham Schroeder starting from a local run of the responder role.

As a side note, notice that there is no attack on Needham Schroeder if we start from the initiator. That is, if we assume that an initiator A completes a run of the protocol intended to be with B, and furthermore the private keys K_A^{-1} and K_B^{-1} of A and B respectively are not known to the penetrator, then it must be the case that a matching run of responder B takes place. In contrast, the attack shown in 11 involves an initiator who intends to communicate with some B' having a compromised private key $K_{B'}^{-1}$. Hence, we do not consider it an attack on an initiator with reasonable key secrecy assumptions.

The NS Shape.
Recall that we are considering Needham-Schroeder from the perspective of a responder B intending to respond to A. Let us suppose that B's nonce N_b has been freshly chosen and that the private keys K_A^{-1} and K_B^{-1} of A and B are both uncompromised, and that B has executed the strand shown at the right in Fig. 9. In protocols using

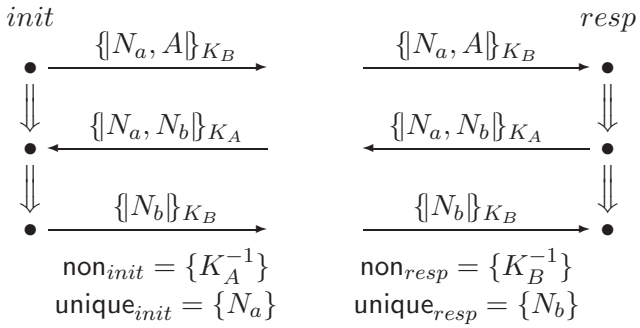

Fig. 8. The Needham Schroeder Protocol

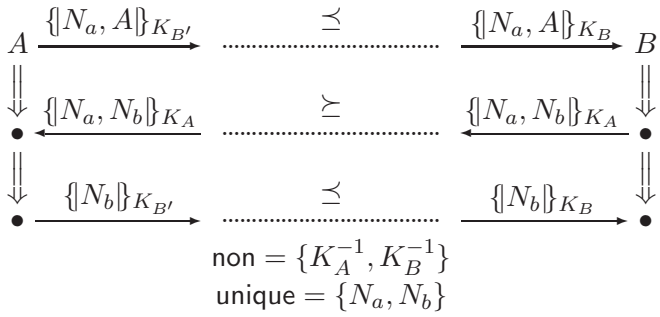

Fig. 9. Only Shape for Needham Schroeder Responder

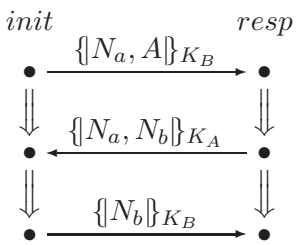

Fig. 10. Needham Schroeder Intended Run

asymmetric encryption, the private keys are used only by recipients to destructure incoming messages. Given that—on a particular occasion—B received and sent these messages, what must have occurred elsewhere in the network?

A must have had a partially matching strand, with the messages sent and received in the order indicated by the arrows of both kinds and the connecting symbols $\preceq$. These symbols mean that the endpoints are ordered, but that other behavior may intervene, whether adversary strands or regular strands. A's strand is only partially matching, because the principal A meant to contact is some B' which may or may not equal B. There is no alternative: Any diagram containing the responder strand of Fig. 9 must contain at least an instance of the initiator strand, with the events ordered as shown, or it cannot have happened.

Although there is a single shape, there are two ways that this shape may be

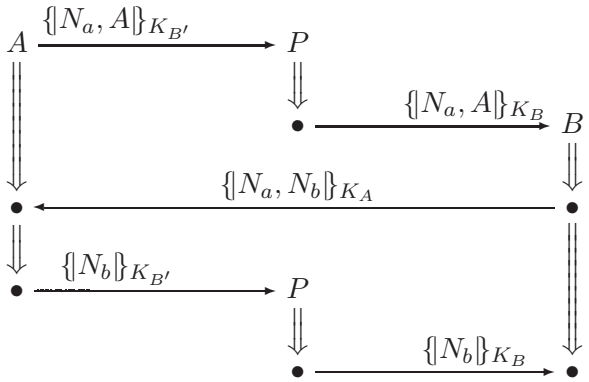

Fig. 11. Bundle: Man-in-the-Middle attack on Needham-Schroeder

realized in executions. Either (1) $B' \neq B$ and hence B''s private key may be compromised, in which case we may complete this diagram with adversary activity to obtain the Lowe attack [6]; or else (2) $B' = B$, leading to the intended run. Note that the shape expresses the most general scenario: there is some B' and we do not know whether or not it is equal to B.

Some protocols have more than one shape. ISO reject has two as we saw above. Otway-Rees has four. Recall that when searching for shapes we started from an initial "skeleton", often just a single strand representing the "point of view" of the analysis.

7.2 Formalizing Shapes

Due to the existence of an infinite set of homomorphisms realizing any preskeleton $\mathbb{A}$, it is useful to find a small subset of those that characterizes all the runs. The notion of characterization is catered to an intuitive understanding of what it means for a protocol run to be an *extension* of another protocol run. Here we capture one notion of extension that we have, in our experience, found to be the most natural and useful.

Definition 7.1 A bundle $\mathcal{B}_2$ *extends* a bundle $\mathcal{B}_1$ if there is an atom replacement (note: not necessarily injective) α such that $\mathcal{B}_1 \cdot \alpha$ is a sub-bundle of $\mathcal{B}_2$. Likewise, for [pre]skeletons: A [pre]skeleton $\mathbb{A}_2$ *extends* a [pre]skeleton $\mathbb{A}_1$ if there is a replacement α such that $\mathbb{A}_1 \cdot \alpha$ is a sub-[pre]skeleton of $\mathbb{A}_2$. Naturally extending this to homomorphisms: A homomorphism $H_2 : \mathbb{A} \to \mathbb{A}_2$ *extends* a homomorphism $H_1 : \mathbb{A} \to \mathbb{A}_1$ if there is an atom replacement α such that $\alpha \circ H_1 : \mathbb{A} \to \mathbb{A}_1 \cdot \alpha$ is simply a codomain restriction of H_2. (i.e. $\mathbb{A}_1 \cdot \alpha$ is a sub-[pre]skeleton of $\mathbb{A}_2$.)

This notion of extension can be equivalently, and more formally, stated as follows:

Definition 7.2 [Pre]Skeleton $\mathbb{A}_2$ extends [pre]skeleton $\mathbb{A}_1$ if and only if there is a node-injective homomorphism from $\mathbb{A}_1$ to $\mathbb{A}_2$.

Homomorphism H_2 extends homomorphism H_1 iff $H_2 = G \circ H_1$ where G is a node-injective homomorphism.

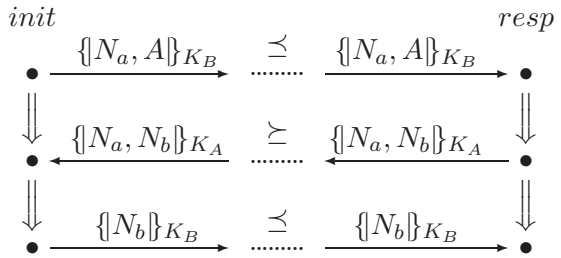

Fig. 12. The Needham Schroeder Protocol Intended Run Skeleton

Example 7.3 The intended-run skeleton for the Needham Schroeder responder in Fig. 12 extends the shape depicted in Fig. 9.

Now we can define what it means for a set of protocol runs (realized homomorphisms/skeletons) to be a characterization for $\mathbb{A}$.

Definition 7.4 A set X of homomorphisms is a characterization for $\mathbb{A}$ if

- Each $H \in X$ realizes $\mathbb{A}$
- Every homomorphism H' realizing $\mathbb{A}$ extends some $H \in X$

One may also think of a characterization for $\mathbb{A}$ as a set of realized skeletons (the set of codomains of the homomorphisms). However, regarding a characterization as a set of realizing homomorphisms from $\mathbb{A}$ is more informative, as it specifies how $\mathbb{A}$ is part of the protocol runs. We may refer to "skeletons in the characterization", in which case we are referring to the codomains of the homomorphisms in the characterization.

Clearly the set of all realizing homomorphisms for $\mathbb{A}$ is a characterization for $\mathbb{A}$. However, we seek a characterization as small as possible: a *minimal characterization*, under some criterion of size (number of nodes, total size of messages... etc). We will show later that there is a single characterization that is a unique minimum for all these definitions of size simultaneously.

This notion of extension described above captures what it means for one protocol run to simply be an elaboration on another run. One can make an argument for several alternative notions of *extends*, including not requiring node-injectivity for G in definition 7.2 above, and/or requiring atom-injectivity of G. Any of these definitions would yield characterizations that can be used to decide security predicates. However, they differ in their size, the ease by which they can be algorithmically constructed, and the ease by which an analyst can interpret the set of all runs and make judgements based on them. We chose this notion of *extends* based on our experience in automating protocol analysis, and we found that the minimum characterizations it yields succinctly capture the intuitive notion of "representative set of all protocol runs".

Requiring node-injectivity in our definition of *extends* has another advantage, in that it results in a partial order as opposed to a preorder. Furthermore, the partial order is well-founded.

Definition 7.5 For skeletons, $\mathbb{A}_1 \leq_n \mathbb{A}_2$ iff $\mathbb{A}_2$ extends $\mathbb{A}_1$. Likewise for homomor-

phisms, $H_1 \leq_n H_2$ iff H_2 extends H_1.

Proposition 7.6 *$\leq_n$ is a well-founded partial order on skeletons (up to isomorphism).*

Proof. Clearly $\leq_n$ is reflexive and transitive.
Antisymmetry: if $\mathbb{A} \leq_n \mathbb{A}'$ via $H = [\phi, \alpha]$ and $\mathbb{A}' \leq_n \mathbb{A}$ via H' then both have the same number of nodes. By considering the composition of H and H', we can see that H is node-bijective (ϕ is bijective) and atom-bijective (α is bijective). By considering the composition of H and H', we can also see that α is bijective from $\mathsf{non}_{\mathbb{A}}$ to $\mathsf{non}_{\mathbb{A}'}$, and also from $\mathsf{unique}_{\mathbb{A}}$ to $\mathsf{unique}_{\mathbb{A}'}$. Likewise ϕ is a bijection from the pairs in $\preceq_{\mathbb{A}}$ to $\preceq_{\mathbb{A}'}$. Therefore $H^{-1} = [\phi^{-1}, \alpha^{-1}]$ is a homomorphism, and is the inverse of H. H is an isomomorphism.
Well-foundedness: For a skeleton $\mathbb{A}$, let $Occ(\mathbb{A})$ be the number of atom occurences in $\mathbb{A}$. Let $Atoms(\mathbb{A})$ be the number of distinct atoms in $\mathbb{A}$, and define the atom-redundancy of a skeleton as $red(\mathbb{A}) = Occ(\mathbb{A}) - Atoms(\mathbb{A})$. Also, let $nonOcc(\mathbb{A})$ and $uniqOcc(\mathbb{A})$ be the number of occurences in $\mathbb{A}$ of atoms in $\mathsf{non}_{\mathbb{A}}$ and $\mathsf{unique}_{\mathbb{A}}$, respectively. We can show that $|\mathsf{node}|$, red , $nonOcc$, $uniqOcc$ and $| \preceq |$ are each non-decreasing with $\leq_n$ and lower bounded by 0. Let

$$Rank(\mathbb{A}) = |\mathsf{node}_{\mathbb{A}}| + red(\mathbb{A}) + nonOcc(\mathbb{A}) + uniqOcc(\mathbb{A}) + | \preceq_{\mathbb{A}} |$$

The rank of a skeleton is non-decreasing with $\leq_n$ and lower bounded by 0. Furthermore, notice that if $\mathbb{A} \leq_n \mathbb{A}'$, and $Rank(\mathbb{A}) = Rank(\mathbb{A}')$, then $\mathbb{A}$ and $\mathbb{A}'$ must be isomorphic. It follows that $\leq_n$ is well-founded up to isomorphism.

□

This extends analogously to homomorphisms.

Corollary 7.7 *$\leq_n$ is a well-founded partial order on homomorphisms (up to isomorphism)*

We will define shapes as the minimal realizing homomorphisms in this well-founded partial order

Definition 7.8 A homomorphism H is a *shape* for $\mathbb{A}$ if H realizes $\mathbb{A}$ and H is minimal under $\leq_n$ amongst homomorphisms realizing $\mathbb{A}$. Let $shapes(\mathbb{A})$ be the set of distinct (up to isomorphism) shapes of $\mathbb{A}$.

Next, it becomes apparent that that the set of shapes of $\mathbb{A}$ describes (can be extended to) all runs compatible with $\mathbb{A}$

Proposition 7.9 *$shapes(\mathbb{A})$ is a characterization for $\mathbb{A}$*

Proof. Since $\leq_n$ is well-founded, for any H' realizing $\mathbb{A}$ there is an $H \in shapes(\mathbb{A})$ such that $H \leq_n H'$, and H' extends H. □

In fact, the set of shapes is the minimum such set

Proposition 7.10 *$shapes(\mathbb{A})$ is a subset of any other characterization for $\mathbb{A}$. Hence $shapes(\mathbb{A})$ is the minimum characterization for $\mathbb{A}$.*

Proof. Take any other characterization X for $\mathbb{A}$. Take any shape $H \in shapes(\mathbb{A})$. Since X is a characterization, there must be some $G \in X$ such that $G \leq_n H$. Since H is minimal under $\leq_n$ then $G \simeq H$.

□

From 7.10 we can see that $shapes(\mathbb{A})$ is the smallest characterization for $\mathbb{A}$ under most definitions of size (number of nodes, total length of messages... etc).

8 Conclusions and Future Work

In cases where the set of shapes is finite, the shapes for a preskeleton $\mathbb{A}$ form a succint representation of all protocol runs compatible with $\mathbb{A}$. In most protocols we have studied, we have found experimentally that any starting preskeleton yields a finite set of shapes. However, it would be useful to identify a subclass of protocols for which this is guaranteed. We expect that a subclass of protocols similar to those defined in [1] and [8] has this property.

We also developed an algorithm, discussed in [2], for constructing the set of shapes for a preskeleton. At a high level, the algorithm starts with the initial preskeleton, finds and solves an authentication test (see [4]) to yield a finite set of solution homomorphisms/preskeletons, then recurses on those. This results is a tree of preskeletons with the shapes at the leaves. These shapes are annotated with secrecy information. The algorithm reasons about secrecy by trying to construct a shape that discloses a value. Further analysis of the properties of this algorithm will be the subject of future work.

References

[1] Bruno Blanchet and Andreas Podelski. Verification of cryptographic protocols: Tagging enforces termination. In Andrew D. Gordon, editor, *Foundations of Software Science and Computation Structures*, number 2620 in LNCS, pages 136–152. Springer, April 2003.

[2] Shaddin F. Doghmi, Joshua D. Guttman, and F. Javier Thayer. Searching for shapes in cryptographic protocols. In *Tools and Algorithms for Construction and Analysis of Systems (TACAS)*, LNCS. Springer, March 2007. Extended version at URL:http://eprint.iacr.org/2006/435.

[3] Joshua D. Guttman. Security goals: Packet trajectories and strand spaces. In Roberto Gorrieri and Riccardo Focardi, editors, *Foundations of Security Analysis and Design*, volume 2171 of *LNCS*, pages 197–261. Springer Verlag, 2001.

[4] Joshua D. Guttman and F. Javier Thayer. Authentication tests and the structure of bundles. *Theoretical Computer Science*, 283(2):333–380, June 2002.

[5] Joshua D. Guttman, F. Javier Thayer, and Lenore D. Zuck. The faithfulness of abstract protocol analysis: Message authentication. *Journal of Computer Security*, 12(6):865–891, 2004.

[6] Gavin Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Proceeedings of* TACAS, volume 1055 of *Lecture Notes in Computer Science*, pages 147–166. Springer Verlag, 1996.

[7] Roger Needham and Michael Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12), 1978.

[8] R. Ramanujam and S. P. Suresh. A decidable subclass of unbounded security protocol. In R. Gorrieri, editor, *WITS '03: Workshop on Issues in the Theory of Security*, pages 11–20, Warsaw, April 2003.

[9] Dawn Xiaodong Song. Athena: a new efficient automated checker for security protocol analysis. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, June 1999.