Completeness of the Authentication Tests*

Shaddin F. Doghmi, Joshua D. Guttman, and F. Javier Thayer

The MITRE Corporation

Abstract. Protocol participants manipulate values, transforming the cryptographic contexts in which they occur. The rules of the protocol determine which transformations are permitted. We formalize these transformations, obtaining new versions of the two *authentication tests* from earlier strand space papers.
We prove that the new versions are *complete*, in this sense: any collection of behaviors that satisfies those two authentication tests, when combined with some feasible adversary behavior, yields a possible execution.
We illustrate the strengthened authentication tests with brief analyses of three protocols.

1 Introduction

Cryptographic protocols are designed to control the ways that protocol participants transform messages. The protocol determines when a critical value may be transmitted within new forms of message. If the critical value has so far occurred only within a particular set of cryptographic contexts, then a participant may be *authenticated* by the way she transforms it to occur in a new context. A protocol preserves *secrecy* by ensuring that no participant's transformation will remove it from a class of safe contexts.

Protocol analysis within a simple Dolev-Yao model [6] may be completely formalized in terms of this idea.

In this paper, we support this assertion, using two forms of the transformation principle. One form covers the case in which the critical value is a fresh, unguessable value such as a nonce or session key. The other covers the case in which the critical value is an encrypted message. Each is a strengthened *authentication test*, various versions of which have appeared in earlier papers [10–12, 14]. We illustrate the different aspects of the strengthened authentication tests in reference to a protocol due to Perrig and Song [14], Yahalom's protocol [3], and a new protocol we call the ambassador's protocol. The authentication tests are sensitive only to the regular (non-adversary) protocol behavior and a set of values assumed uncompromised; they are insensitive to specific adversary behavior.

We work within the strand space model [11], so local behaviors of regular principals are represented by regular strands, and adversary behavior is represented by penetrator strands. Possible executions are represented by bundles. (See Definitions 2–5.)

* Supported by the National Security Agency and by MITRE-Sponsored Research.
Addresses: shaddin@stanford.edu, {guttman, jt}@mitre.org. Appears as [4].

Our main result is *completeness* for the authentication tests, in the following sense. Suppose that a collection of regular strands has been chosen, as well as a collection X of values that we assume the adversary did not originally possess and will not guess. Then there exists a bundle $\mathcal{B}$ containing exactly the given strands, without the adversary using the values X, if and only if those strands and the values X satisfy the authentication tests (Prop 5).

An implementation called CPSA uses most aspects of the strengthened authentication tests. It searches for all the minimal, essentially different executions that a protocol allows, as described in [5]. We call these minimal, essentially different executions *shapes*. CPSA checks authentication and secrecy properties, since these are easily read off from the set of shapes. By the undecidability of these properties [7], there exist protocols for which the set of shapes is infinite; however, for many protocols the set of shapes is very small, and frequently only one or two.

Related work. There is a vast body of work on protocol analysis. Much, such as Cryptyc [8] and ProVerif [1], aim at sound but not complete methods. Others, for instance Athena [14], do a search involving both regular and adversary behaviors. We here propose a method that is complete in the sense we have mentioned, but considers adversary behavior only in the most limited way. In particular, the authentication tests consider only whether given values—generally, keys—are available to the adversary, but not what actions are needed to synthesize the values received by the regular participants.

Roadmap to this paper. In Section 2 we give the basic strand space definitions, adapted to our current needs. Section 3 summarizes three examples that together illustrate many aspects of protocol analysis. Section 4 defines the key idea of a message "occurring only within" certain contexts in another message; we also provide a number of examples that will be used later in the paper. Section 5 gives the authentication test principles, and illustrates how to use them to analyze our three examples. Section 6 considers the adversary in more detail, and gives the crucial lemma for completeness. Finally, Section 7 introduces the notion of skeleton and uses it to formalize completeness. Appendix A fills in the proof of the key lemma.

2 Terms, Strands, and Bundles

In this section, we give background definitions, which are somewhat more general than those in the extended version of [5].

2.1 Algebra of Terms

Terms (or messages) form a free algebra A, built from (typed) atoms and (untyped) indeterminates $g, h, \ldots$ via constructors.

The atoms may be partitioned into some types, e.g. *keys*, *nonces*, etc. We assume A contains infinitely many atoms of each type.

An inverse operator is defined on atomic keys. There may be additional functions on atoms, such as an injective *public key of* function mapping principals to keys, or an injective *long term shared key of* function mapping pairs of principals to keys. These functions are not constructors, and their results are atoms. For definiteness, we include here functions $\mathsf{pubk}(a), \mathsf{ltk}(a)$ mapping principals to (respectively) their public keys and to a symmetric key shared on a long-term basis with a fixed server S. $\mathsf{pubk}(a)^{-1}$ is a's private key, where $\mathsf{pubk}(a)^{-1} \neq \mathsf{pubk}(a)$. By contrast, $\mathsf{ltk}(a)^{-1} = \mathsf{ltk}(a)$.

Terms in A are freely built from atoms and indeterminates using *tagged concatenation* and *encryption*. The tags are chosen from a set of constants written in sans serif font (e.g. tag). The tagged concatenation using tag of t_0 and t_1 is written $\mathsf{tag}\,\hat{}\,t_0\,\hat{}\,t_1$. Tagged concatenation using the distinguished tag null of t_0 and t_1 is written $t_0\,\hat{}\,t_1$.

Encryption takes a term t and a term t' serving as a key, and yields a term as result written $\{|t|\}_{t'}$. Protocols generally use a term t' as a key only if it is of some special forms, such as an atomic key or a term produced by hashing (used as a symmetric key). We regard hashing as encryption with a public key the inverse of which is not known to any principal. We extend the inverse function to non-atomic keys by stipulating that if K is non-atomic, then $K^{-1} = K$. We write $\{|t|\}_K$ to cover both the case of atomic and non-atomic K.

We regard terms as abstract syntax trees, where atoms and indeterminates are the leaves. A concatenation $\mathsf{tag}\,\hat{}\,t_0\,\hat{}\,t_1$ has a root node labeled tag and the two immediate subtrees representing t_0, t_1. An encryption $\{|t_0|\}_{t_1}$ has a root labeled with t_1, and one immediate subtree representing t_0.

Replacements are essentially homomorphisms on the algebra A:

Definition 1 (Replacement, Application). A *replacement* is a function α mapping atoms and indeterminates to 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., $\alpha(K^{-1}) = (\alpha(K))^{-1}$ and $\alpha(\mathsf{pubk}(a)) = \mathsf{pubk}(\alpha(a))$.

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

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

We extend the homomorphism $\_ \cdot \alpha$ to larger objects such as pairing and sets; thus, $(x, y) \cdot \alpha = (x \cdot \alpha, y \cdot \alpha)$, and $S \cdot \alpha = \{x \cdot \alpha \colon x \in S\}$. If $x \notin \mathsf{A}$ is a simple value such as an integer or a symbol, then $x \cdot \alpha = x$.

2.2 Strands and Origination

Directed messages represent transmission and reception of messages, where the direction $+$ means transmission, and the direction $-$ means reception:

Definition 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 $(s, \alpha) \mapsto s \cdot \alpha$ such that $\mathsf{tr}(s \cdot \alpha) = (\mathsf{tr}(s)) \cdot \alpha$.

By a *strand*, we just mean any member of some strand space Σ.

Definition 3. A *penetrator strand* has trace of one of the following forms:

M_t: $\langle +t \rangle$ where $t \in$text, principal,nonce
K_K: $\langle +K \rangle$ with atomic key K
$\mathsf{C}_{g,h}$: $\langle -g,\ -h,\ +g\,\hat{}\,h \rangle$
$\mathsf{S}_{g,h}$: $\langle -g\,\hat{}\,h,\ +g,\ +h \rangle$
$\mathsf{E}_{h,K}$: $\langle -K,\ -h,\ +\{\!|h|\!\}_K \rangle$
$\mathsf{D}_{h,K}$: $\langle -K^{-1},\ -\{\!|h|\!\}_K,\ +h \rangle$.

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

The *subterm* relation, written $\sqsubseteq$, is the least reflexive, transitive relation such that (1) $t_0 \sqsubseteq \mathsf{tag}\,\hat{}\,t_0\,\hat{}\,t_1$; (2) $t_1 \sqsubseteq \mathsf{tag}\,\hat{}\,t_0\,\hat{}\,t_1$; and (3) $t \sqsubseteq \{\!|t|\!\}_K$. Notice, however, $K \not\sqsubseteq \{\!|t|\!\}_K$ unless (anomalously) $K \sqsubseteq t$. The subterms of t are the terms represented by the subtrees of t's abstract syntax tree. We say that a key K is *used for encryption* in a term t if for some t_0, $\{\!|t_0|\!\}_K \sqsubseteq t$.

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 prefer to write $s \downarrow i$ for the node $n = (s, i)$. A term t *originates* at node n if n is positive, $t \sqsubseteq \mathsf{msg}(n)$, and $t \not\sqsubseteq \mathsf{msg}(m)$ 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.

2.3 Protocols and Bundles

Definition 4 (Protocols). A *candidate* $\langle \Pi, \mathsf{strand\_non}, \mathsf{strand\_unique} \rangle$ consists of: (1) a finite set Π of strands called the *roles* of the protocol; (2) a function $\mathsf{strand\_non}$ mapping each role r to a finite set of keys $\mathsf{strand\_non}_r$, the non-originating keys of r; and (3) a function $\mathsf{strand\_unique}$ mapping each role r to a finite set of atoms $\mathsf{strand\_unique}_r$, the uniquely originating atoms of r.

A candidate $\langle \Pi, \mathsf{strand\_non}, \mathsf{strand\_unique} \rangle$ is a *protocol* if (1) $K \in \mathsf{strand\_non}_r$ implies that K does not occur in any node of r, but either K or K^{-1} is used for encryption on some term of $\mathsf{tr}(r)$; and (2) $a \in \mathsf{strand\_unique}_r$ implies that a originates on r.

The *regular strands* of $\langle \Pi, \mathsf{strand\_non}, \mathsf{strand\_unique} \rangle$ form the set of instances of the roles, $\Sigma_\Pi = \{r \cdot \alpha : r \in \Pi\}$.

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.

Definition 5 (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) when $n_2 \in \mathcal{N}_\mathcal{B}$ is negative, there is exactly one $n_1 \in \mathcal{N}_\mathcal{B}$ 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 $(\rightarrow_{\mathcal{B}} \cup \Rightarrow_{\mathcal{B}})$.

$\mathcal{B}$ is a *bundle over* $\langle \Pi, \mathsf{strand\_non}, \mathsf{strand\_unique} \rangle$ 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 $a \in \mathsf{strand\_non}_r \cdot \alpha$, then a does not occur in $\mathcal{B}$; and (3) if $s = r \cdot \alpha$ and $a \in \mathsf{strand\_unique}_r \cdot \alpha$, then a originates at most once in $\mathcal{B}$.

We say that a strand s is *in* $\mathcal{B}$ if s has at least one node in $\mathcal{B}$. Henceforth, assume fixed some arbitrary protocol $\langle \Pi, \mathsf{strand\_non}, \mathsf{strand\_unique} \rangle$.

Proposition 1. *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 $a \sqsubseteq \mathsf{msg}(n)$ for any $n \in \mathcal{B}$, then a originates at some $m \preceq_{\mathcal{B}} n$.*

If α is a replacement, and $\mathcal{B}$ is a bundle, then $\mathcal{B} \cdot \alpha$ is a bundle.

$\mathcal{B} \cdot \alpha$ is a bundle *over* the same protocol if the replacement α does not cause the origination assumptions to fail.

3 Some Example Protocols

We consider here three relevant protocol examples.

Example 1 (Perrig-Song). The PS protocol (Fig. 1) is due to Perrig and Song, or rather, invented by their automated protocol generator [14]. Here, A and B share a long-term symmetric key, and the purpose of the protocol is to provide mutual authentication using the key. If the nonces are chosen to be Diffie-Hellman values, i.e. $N_a = g^x, N_b = g^y$, then the participants can combine those values to obtain an authenticated shared secret g^{xy} at the end.

The authors mention a reflection attack if B's name is omitted from the second message.

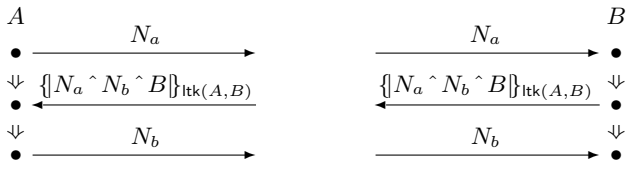

Fig. 1. PS symmetric key protocol

Example 2 (Yahalom). The Yahalom protocol [3] is also a symmetric key protocol, in this case using a key server to generate a session key whose reception by A is confirmed to B in the protocol (Fig. 2). This clever and compact protocol uses a surprising range of the tricks of protocol analysis.

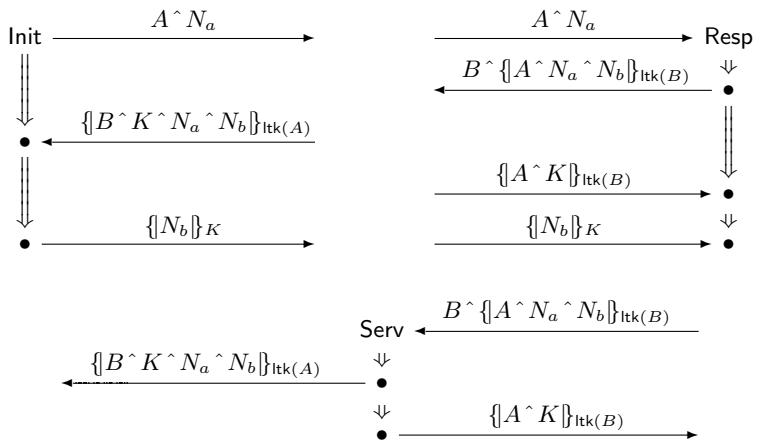

Fig. 2. Yahalom protocol (forwarding removed)

Example 3 (The Ambassador's Protocol). A new protocol, which we call the ambassador's protocol, illustrates a rarely used aspect of the complete authentication tests. In this protocol (Fig. 3), a government G delivers a signed and encrypted authorization to its ambassador A. If negotiations are successful, then the ambassador performs the decryption and delivers the commitment to the foreign government F, which countersigns and returns the now reciprocal commitment to G. Typically, G would perform its first step with many potential messages m; after negotiations, the ambassador would select an appropriate session to complete. It is important that the commitments are encrypted so that A's portfolio of negotiating strategies is not disclosed to F. The fact that a particular commitment is decrypted tells G that the negotiations completed with this outcome.

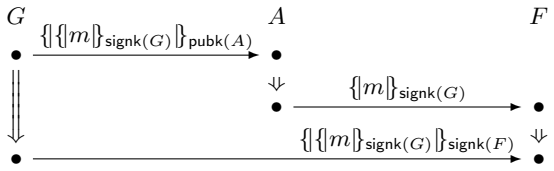

Fig. 3. The Ambassador's Protocol

4 "Occurs Only Within"

The most important idea for stating the strengthened authentication tests is of a term t_0 occurring only within specific forms in some other term.

Suppose that a set $S = \{\{|t_0|\}_{K_0}, \{|t_1|\}_{K_1}, \dots\}$ contains only encryptions. We say that a term t occurs only within S in t' if every path through the abstract syntax tree of t' that ends with t traverses a member of S.[1] The recursive formulation of Definition 6 is equivalent.

When $t \not\sqsubseteq t'$, then no path reaches it, so t occurs only within S in t' for every set of encryptions S, vacuously.

Definition 6 (Occurs only within/outside). Let S be a set of encryptions. A term t_0 *occurs only within* S in t, if:

1. $t_0 \not\sqsubseteq t$; or
2. $t \in S$; or
3. $t \neq t_0$, and either (3a) $t = \{|t_1|\}_K$ and t_0 occurs only within S in t_1, or else (3b) $t = \mathsf{tag}\,\hat{}\,t_1\,\hat{}\,t_2$ and t_0 occurs only within S in each t_i $(i = 1, 2)$.

It *occurs outside* S *in* t if t_0 does not occur only within S in t.

We say that t *has exited* S *passing from* t_0 *to* t_1 if t occurs only within S in t_0 but t occurs outside S in t_1. Term t *exits* S *at* a node n if t occurs outside S in $\mathsf{msg}(n)$ but occurs only within S in every $\mathsf{msg}(m)$ for $m \prec n$.

If it occurs outside S, this means that $t_0 \sqsubseteq t$ and there is a non-empty path from the root to an occurrence of t_0 as a subterm of t that traverses no $t_1 \in S$. "Occurring only within" is similar to "being protected by a set of hat-terms" [2].

Example 4 (PS Occurrences). N_b occurs only within the singleton set

$$S_{ps} = \{\{|N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(A,B)}\}$$

in the term $\{|N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(A,B)}$. It has exited S_{ps} passing from $\{|N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(A,B)}$ to N_b. This provides the responder's guarantee.

$\{|N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(A,B)}$ occurs only within the null set $\emptyset$ in N_a, that is, it does not occur at all in N_a. However, this encryption occurs outside $\emptyset$ in $\{|N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(A,B)}$. This provides the initiator's guarantee.

Example 5 (Yahalom Occurrences). N_b exits $S_{Y,1} = \{\{|A\,\hat{}\,N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(B)}\}$ passing from $\{|A\,\hat{}\,N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(B)}$ to the server's output $\{|B\,\hat{}\,K\,\hat{}\,N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(A)}$. The nonce N_b exits the larger set

$$S_{Y,2} = \{\{|A\,\hat{}\,N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(B)}\} \cup \{\{|B\,\hat{}\,K''\,\hat{}\,N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(A)} : K'' \text{ is a key }\}$$

when passing from $\{|A\,\hat{}\,N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(B)}$ to any term of the form $\{|N_b|\}_{K'}$. When $K' \neq K$, N_b has exited $S_{Y,2} \cup \{\{|N_b|\}_{K'}\}$ when passing from $\{|A\,\hat{}\,N_a\,\hat{}\,N_b|\}_{\mathsf{ltk}(B)}$ to $\{|N_b|\}_K$.

The correctness of the protocol, for the responder, relies on these three steps.

[1] In our terminology (Section 2), the K in $\{|t|\}_K$ is not an occurrence as a subterm, and no path in the syntax tree reaches it.

Example 6 (Ambassador's Protocol). The signed message $\{|m|\}_{\mathsf{signk}(G)}$ occurs outside the empty set $\emptyset$ in $\{|m|\}_{\mathsf{signk}(G)}$.

It has exited $S_a = \{\{|\{|m|\}_{\mathsf{signk}(G)}|\}_{\mathsf{pubk}(A)}\}$ when passing from the term $\{|\{|m|\}_{\mathsf{signk}(G)}|\}_{\mathsf{pubk}(A)}$ to $\{|m|\}_{\mathsf{signk}(G)}$.

5 The Strengthened Authentication Tests

When a principal follows the rules of a protocol, it transforms the way that a critical value occurs in messages. A critical value that has hitherto occurred only within a limited set of forms is freed from them, and retransmitted in a new form. The outgoing and incoming authentication tests describe the possible executions of protocols using this idea. The outgoing test deals with the case where the critical value is a uniquely originating atom, and the incoming test deals with the case where it is an encryption.

5.1 The Outgoing Authentication Test

We say that t is *disclosed* in $\mathcal{B}$ iff $\mathsf{msg}(n) = t$ for some $n \in \mathcal{B}$. By the definitions of the penetrator strands for encryption and decryption (Definition 3), if the adversary uses K for encryption or decryption anywhere in $\mathcal{B}$, then K is disclosed in $\mathcal{B}$. If K^{-1} is not disclosed, it cannot decrypt any term encrypted with K.

We say that t is *disclosed before* m in $\mathcal{B}$, iff, for some $n \in \mathcal{B}$, $\mathsf{msg}(n) = t$ and $n \prec_{\mathcal{B}} m$. If a key is not disclosed before a negative node m, then the adversary cannot use that key to prepare the term received on m.

Proposition 2 (Outgoing Authentication Test). *Suppose an atom a originates uniquely at a regular node n_0 in bundle $\mathcal{B}$, and suppose for some $n_1 \in \mathcal{B}$, a has exited S passing from $\mathsf{msg}(n_0)$ to $\mathsf{msg}(n_1)$, where S is a set of encryptions.*

Then either (1) there exists some $\{|t|\}_K \in S$ such that K^{-1} is disclosed before n_1 in $\mathcal{B}$, or else (2) a exits from S at *some positive* regular *$m_1 \preceq_{\mathcal{B}} n_1$. If in case (2) n_0 and m_1 lie on different strands, then for some negative $m_0 \in \mathcal{B}$ with $a \sqsubseteq \mathsf{msg}(m_0)$,*

$$n_0 \prec_{\mathcal{B}} m_0 \Rightarrow^+ m_1 \preceq_{\mathcal{B}} n_1.$$

Proof. Suppose, contrary to case (1), that no K^{-1} for $\{|t|\}_K \in S$ is disclosed before n_1 in $\mathcal{B}$. Apply Prop. 1 to $T =$

$$\{m\colon m \preceq_{\mathcal{B}} n_1 \text{ and } a \text{ occurs outside } S \text{ in } \mathsf{msg}(m)\};$$

$n_1 \in T$, so T has $\preceq_{\mathcal{B}}$-minimal members m_1. Since keys K used in S have K^{-1} not disclosed before n_1, m_1 cannot lie on a decryption penetrator D-strand. By unique origination, a does not lie on a M-strand or K-strand. By the definitions of S and "occurs only within," m_1 does not lie on a S-, C-, or E-strand. Thus, m_1 lies on some $s \in \Sigma_\Pi$. If n_0 does not lie on s, then a does not originate on s, so $a \sqsubseteq \mathsf{msg}(m_0)$ for some negative m_0, with $m_0 \Rightarrow^+ m_1$. □

In the outgoing test, we call $m_0 \Rightarrow^+ m_1$ an *outgoing transforming edge* for a, S. It transforms the occurrence of a, causing a to exit S. We call (n_0, n_1) an *outgoing test pair for* a, S when a originates uniquely at n_0 and a has exited S passing from $\mathsf{msg}(n_0)$ to $\mathsf{msg}(n_1)$. We also sometimes call m_1 an *outgoing transforming node* and n_1 an *outgoing test node*.

Example 7. In the Perrig-Song protocol, with responder role s_r, the nodes $(s_r \downarrow 2), (s_r \downarrow 3)$ form an outgoing test pair for N_b, S_{ps}, where S_{ps} is as given in Example 4.

The initiator role s_i has the only outgoing transforming edge for N_b, S_{ps}, lying on $s_i \downarrow 2 \Rightarrow s_i \downarrow 3$. Hence, if any bundle $\mathcal{B}$ has uncompromised long term key $\mathsf{ltk}(A, B)$, and $\mathcal{B}$ contains the three nodes of any responder strand, then $\mathcal{B}$ also contains the three nodes of an initiator strand with matching parameters.

This is the responder's authentication result.

Many protocols can be verified using only singleton sets like S_{ps}, and this was the part of the outgoing authentication test given in [11, 9]. However, there are other protocols in which the same critical value is transformed more than once, and these protocols cannot be verified using only singleton sets S. For instance, in the Yahalom protocol, the responder's nonce N_b is transformed first by the key server and then again by the initiator. To verify the presence of the initiator,

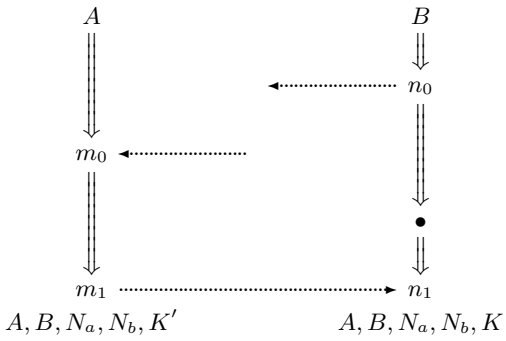

Fig. 4. Yahalom bundle containing responder

we use a set that includes the original form transmitted by the responder, and also all forms that could be produced from it by means of a server strand. To cause N_b to escape from this set $S_{Y,2}$, a server strand cannot suffice: we need an initiator strand.

Example 8 (Yahalom: Inferring Initiator). As in Example 5, letting

$$S_{Y,2} = \{\{\!|A \,\hat{}\, N_a \,\hat{}\, N_b|\!\}_{\mathsf{ltk}(B)}\} \cup \{\{\!|B \,\hat{}\, K'' \,\hat{}\, N_a \,\hat{}\, N_b|\!\}_{\mathsf{ltk}(A)} : K'' \text{ is a key }\},$$

by Prop. 2, if $\mathcal{B}$ contains a full Yahalom responder strand, then either one of the keys $\mathsf{ltk}(A), \mathsf{ltk}(B)$ is compromised, or else there is an initiator strand in $\mathcal{B}$ agreeing on A, B, N_a, N_b, although possibly not on the session key K' (Fig. 4). Another application of Prop. 2 allows us to interpolate a server run into the middle column of Fig. 4, as shown in Fig 5. We instantiate n_0, n_1 from the theorem by the nodes labeled n_0 and n_1' in Fig. 5. This application uses the singleton set $S_{Y,1}$ from Example 5.

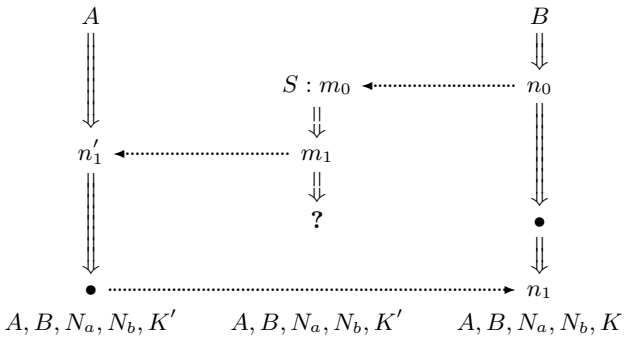

Fig. 5. Yahalom bundle containing responder and server

The outgoing authentication test is also the main theorem for establishing secrecy for session keys and other values that are transmitted in protocols. Long-term keys, which are typically used but never transmitted in any form in protocols, are typically secret only by assumption. However, the outgoing test allows us to infer that a session key—such as the one in the Yahalom protocol—will remain secret assuming that the participants' long-term keys are uncompromised.

Example 9 (Yahalom Session Key Secrecy). Suppose that $\mathcal{B}$ is a bundle in which $\mathsf{ltk}(A), \mathsf{ltk}(B)$ are uncompromised, in which K' originates uniquely on a server strand s_s with parameters A, B, N_a, N_b, K'. Then K' is uncompromised in $\mathcal{B}$. The reason is that otherwise, we may apply Prop. 2 to the set

$$S_{Y,3} = \{\{|B \hat{} K' \hat{} N_a \hat{} N_b|\}_{\mathsf{ltk}(A)}, \{|A \hat{} K'|\}_{\mathsf{ltk}(B)}\}.$$

There is no role of the protocol that, having received K' occurring only within $S_{Y,3}$, would retransmit it outside this form. Thus, given that the keys used in $S_{Y,3}$ are assumed uncompromised, we have refuted the assumption that K' could occur compromised in $\mathcal{B}$.

Thus, secrecy relies on the absence of an outgoing transforming edge. We also use the outgoing test negatively to prove that values that otherwise could be different are in fact equal.

Example 10 (Yahalom Session Key Agreement). In Fig. 5, we must in fact have $K' = K$. Otherwise, N_b has exited the set

$$S_{Y,4} = S_{Y,2} \cup \{\{|N_b|\}_{K''} : K'' \neq K\}$$

passing from n_0 to n_1. However, we have assumed that $\mathsf{ltk}(A), \mathsf{ltk}(B)$ are uncompromised, and we have now ascertained that K' is uncompromised also. Thus, there would have to be a outgoing transforming edge for $N_b, S_{Y,4}$, but the Yahalom protocol does not furnish any role that would do so.

Thus, we have illustrated that the outgoing authentication test is a highly versatile protocol analysis tool. It allows repeated use of a nonce for authentication; it helps prove secrecy for values that a protocol distributes; and it allows us to prove equality of values when certain messages cannot be transformed by the protocol.

5.2 The Incoming Authentication Test

The incoming test principle is similar, except that the critical value is an encryption $t = \{|t_0|\}_K$. In this case, the transforming edge may be a single node m_1 that emits t, rather than the pair we have in the outgoing case. The node m_1 is not always preceded by another node m_0 that has received t.

Proposition 3 (Incoming Authentication Test). *Let $t = \{|t_0|\}_K$ and let S be a set of encryptions. If t occurs outside S in any $n_1 \in \mathcal{B}$, then either (1) K is disclosed before n_1 in $\mathcal{B}$, or (2) for some K_0 with $\{|t|\}_{K_0} \in S$, K_0^{-1} is disclosed before n_1 in $\mathcal{B}$, or (3) t exits S at some positive regular $m_1 \preceq_{\mathcal{B}} n_1$.*

Proof Sketch. Apply Prop. 1 to the set $T = \{m : m \preceq_{\mathcal{B}} n_1$ and t occurs outside S in $\mathsf{msg}(m)\}$. □

We call m_1 an *incoming transforming node* for t, S, and n_1 an *incoming test node* for t, S. In our experience with existing protocols, Prop. 3 is almost always used with $S = \emptyset$, i.e. t does not occur at all before m_1. However, one can invent protocols, like the ambassador's protocol, requiring non-empty S, and completeness requires the stronger form. We first illustrate the more usual case $S = \emptyset$.

Example 11 (PS Initiator's Guarantee). Suppose that, in a PS bundle $\mathcal{B}$, A has transmitted N_a and received $\{|N_a \hat{} N_b \hat{} B|\}_{\mathsf{ltk}(A,B)}$. Then $\mathcal{B}$ contains at least the first two nodes of a matching responder strand, unless $\mathsf{ltk}(A, B)$ is compromised in $\mathcal{B}$. To prove this, one applies Prop. 3 to $t = \{|N_a \hat{} N_b \hat{} B|\}_{\mathsf{ltk}(A,B)}$, $S = \emptyset$, and $n_1 =$ the initiator's second node.

One can also use the incoming test in a similar way in the Yahalom protocol (see Fig. 5) to show that the server strand's last node—marked **?** there—has occurred.

Example 12 (Ambassador's Protocol). Let $\mathcal{B}$ be a bundle for the Ambassador's Protocol, and suppose that G's first and second nodes are both contained in $\mathcal{B}$. Then the ambassador A has a full run with the same message m.

To prove this, we apply Prop. 3 with $S = \{\{|\{|m|\}_{\mathsf{signk}(G)}|\}_{\mathsf{pubk}(A)}\}$. The message $\{|m|\}_{\mathsf{signk}(G)}$ has exited from S passing from G's first to G's second node. Thus, either $\mathsf{privk}(A)$ is compromised, or A has extracted $\{|m|\}_{\mathsf{signk}(G)}$ from S.

6 Penetrator Webs and Test Nodes

We can see that Props. 2–3 have some sort of completeness by considering the powers of the adversary. In essence, if any negative regular node is neither an outgoing test node nor an incoming test node, then the adversary can derive the term on it. Thus, only test nodes in this sense can provide authentication guarantees about the presence of regular activity. The rest could be the adversary's work.

To make this precise, we define penetrator webs, which characterize what the adversary can do with fixed inputs from the regular participants.

Definition 7 (Penetrator web, derivable). Let $G = \langle \mathcal{N}_G, (\rightarrow_G \cup \Rightarrow_G) \rangle$ be a finite acyclic subgraph of $\langle \mathcal{N}, (\rightarrow \cup \Rightarrow) \rangle$ such that $\mathcal{N}_G$ consists entirely of penetrator nodes. G is a *penetrator web* with support S_{spt} and result R if S_{spt} and R are sets of terms and moreover:

1. If $n_2 \in \mathcal{N}_G$ is negative, then either $\mathsf{msg}(n_2) \in S_{\mathrm{spt}}$ or there is a unique n_1 such that $n_1 \rightarrow_G n_2$.
2. If $n_2 \in \mathcal{N}_G$ and $n_1 \Rightarrow n_2$ then $n_1 \Rightarrow_G n_2$.
3. For each $t \in R$, either $t \in S_{\mathrm{spt}}$ or for some positive $n \in \mathcal{N}_G$, $\mathsf{msg}(n) = t$.

If V is a set of atoms, then term t_1 is *derivable from* S_{spt} *avoiding* V if there is a web G with support $S_G \subseteq S_{\mathrm{spt}}$ and $t_1 \in R_G$, where no atom in V originates on a penetrator strand in G.

If $n \in \mathcal{B}$ is a negative node, then $\mathcal{B}$ includes a penetrator web G with result $R_G = \{\mathsf{msg}(n)\}$. Its support $S_G = \{\mathsf{msg}(m) \colon m \text{ is positive regular and } m \prec_{\mathcal{B}} n\}$.

When S_{spt} is a set of terms, we say that t *has exited* S_{enc} *passing from* S_{spt} *to* t_1 if for each $t_0 \in S_{\mathrm{spt}}$, t has exited S_{enc} passing from t_0 to t_1. Def. 6 says that this means that t occurs only within the encryptions in S_{enc} in every $t_0 \in S_{\mathrm{spt}}$, and t occurs outside S_{enc} in t_1.

In the following proposition, the first condition says that when $t_1 \neq \mathsf{msg}(n_1)$ is an outgoing test node $n_1 \in \mathcal{B}$, then we do not need to add an outgoing transforming edge. The second condition says that when $t_1 \neq \mathsf{msg}(n_1)$ is an incoming test node $n_1 \in \mathcal{B}$, we do not need to add an incoming transforming node. The conclusion is that the term is then derivable.

Proposition 4. *Let V be a set of atoms; let S_{spt} be a finite set of terms; and let t_1 be a term such that, for any $a \in V$, if $a \sqsubseteq t_1$, then $a \sqsubseteq t_0$ for some $t_0 \in S_{\mathrm{spt}}$. Suppose the following conditions hold:*

1. *for all $a \in V$ and all sets of encryptions S_{enc}, if a has exited S_{enc} passing from S_{spt} to t_1, then there is some $\{\!|t|\!\}_{K_0} \in S_{\mathrm{enc}}$, such that ${K_0}^{-1}$ is derivable from S_{spt} avoiding V; and*
2. *for all encryptions $\{\!|t|\!\}_K$, and all sets of encryptions S_{enc}, if $\{\!|t|\!\}_K$ has exited S_{enc} passing from S_{spt} to t_1, then either K is derivable from S_{spt} avoiding V, or else some ${K_0}^{-1}$ with $K_0 \in \mathsf{used}(S_{\mathrm{enc}})$ is derivable from S_{spt} avoiding V.*

Then term t_1 is derivable from S_{spt} avoiding V.

A proof is in Appendix A. One can easily determine whether t is derivable from S_{spt}, since penetrator webs *normalize* [11, Proposition 5] so that all their destructive steps precede their constructive steps (cf. [13]). Thus, there are only as many intermediate values as there are subterms of $S_{\mathrm{spt}} \cup \{t\}$.

7 Completeness

In order to extract the completeness result from Proposition 4, it is convenient to introduce the notion of *skeleton*, following [5]. A skeleton is potentially the regular (non-penetrator) part of a bundle or of some portion of a bundle. We may regard a bundle as "put together" using a skeleton and one penetrator web for each negative regular node within it.

A skeleton consists of nodes annotated with additional information, indicating order relations among the nodes, uniquely originating atoms, and non-originating atoms. We say that an atom a *occurs* in a set nodes of nodes if for some $n \in \mathsf{nodes}$, $a \sqsubseteq \mathsf{msg}(n)$. A key K is *used* in nodes if for some $n \in \mathsf{nodes}$, $\{\!|t|\!\}_K \sqsubseteq \mathsf{msg}(n)$. We say that a key K is *mentioned in* nodes if K or K^{-1} either occurs or is used in nodes. For a non-key a, a is mentioned if it occurs.

Definition 8. A four-tuple $\mathbb{A} = (\mathsf{nodes}_{\mathbb{A}}, \preceq_{\mathbb{A}}, \mathsf{non}_{\mathbb{A}}, \mathsf{unique}_{\mathbb{A}})$ is a *skeleton* if:

1. $\mathsf{nodes}_{\mathbb{A}}$ is a finite set of regular nodes; $n_1 \in \mathsf{nodes}$ and $n_0 \Rightarrow^+ n_1$ implies $n_0 \in \mathsf{nodes}_{\mathbb{A}}$;
2. $\preceq_{\mathbb{A}}$ is a partial ordering on $\mathsf{nodes}_{\mathbb{A}}$ such that $n_0 \Rightarrow^+ n_1$ implies $n_0 \preceq_{\mathbb{A}} n_1$;
3. $\mathsf{non}_{\mathbb{A}}$ is a set of atomic keys, and for all $K \in \mathsf{non}_{\mathbb{A}}$, either K or K^{-1} is used in $\mathsf{nodes}_{\mathbb{A}}$, and for all $K \in \mathsf{non}_{\mathbb{A}}$, K does not occur in $\mathsf{nodes}_{\mathbb{A}}$;
4. $\mathsf{unique}_{\mathbb{A}}$ is a set of atoms, and for all $a \in \mathsf{unique}_{\mathbb{A}}$, a occurs in $\mathsf{nodes}_{\mathbb{A}}$, and $a \in \mathsf{unique}_{\mathbb{A}}$ implies a originates at no more than one node in $\mathsf{nodes}_{\mathbb{A}}$.

We think of a skeleton as describing a set of bundles; for our present purposes it is enough to consider the bundles into which a skeleton may be embedded:

Definition 9. If $\mathbb{A} = (\mathsf{nodes}_{\mathbb{A}}, \preceq_{\mathbb{A}}, \mathsf{non}_{\mathbb{A}}, \mathsf{unique}_{\mathbb{A}})$ is a skeleton and $\mathcal{B}$ is a bundle, then $\mathbb{A}$ is *embedded in* $\mathcal{B}$ if:

1. For all $n \in \mathsf{nodes}_{\mathbb{A}}$, $n \in \mathcal{B}$;
2. If $n_0 \preceq_{\mathbb{A}} n_1$, then $n_0 \preceq_{\mathcal{B}} n_1$;
3. For all $K \in \mathsf{non}_{\mathbb{A}}$, K originates nowhere in $\mathcal{B}$;

4. For all $a \in \mathsf{unique}_{\mathbb{A}}$, a originates uniquely in $\mathcal{B}$.

The embedding is *tight* if for all regular $n \in \mathcal{B}$, $n \in \mathsf{nodes}_{\mathbb{A}}$, and whenever $n_0, n_1 \in \mathsf{nodes}_{\mathbb{A}}$ and $n_0 \preceq_{\mathcal{B}} n_1$, then $n_0 \preceq_{\mathbb{A}} n_1$.

A message t is *potentially compromised before n in $\mathbb{A}$* if, letting

$$V = \mathsf{non}_{\mathbb{A}} \cup (\mathsf{unique}_{\mathbb{A}} \cap \{a \colon a \text{ originates somewhere in } \mathbb{A}\}),$$

and

$$S_{\mathrm{spt}} = \{\mathsf{msg}(m) \colon m \preceq_{\mathbb{A}} n \text{ and } n \text{ is positive }\},$$

t is derivable from S_{spt} avoiding V. Evidently, $\mathbb{A}$ is tightly embedded in a bundle if, for every negative $n \in \mathbb{A}$, $\mathsf{msg}(n)$ is potentially compromised before n in $\mathbb{A}$.

We regard a skeleton $\mathbb{A}$ as satisfying the authentication test properties when "disclosed before n" is interpreted as meaning potentially compromised before n. That is:

Definition 10. *$\mathbb{A}$ satisfies the outgoing authentication test* if and only if the following is true for all $n_0, n_1 \in \mathbb{A}$, and for all atoms a and sets of encryptions S. If a has exited S passing from n_0 to n_1, then either (1) there exists some $\{\!|t|\!\}_K \in S$ such that K^{-1} is potentially compromised before n_1 in $\mathbb{A}$, or else (2) a exits from S *at* some positive *regular* $m_1 \preceq_{\mathbb{A}} n_1$.

$\mathbb{A}$ *satisfies the incoming authentication test* if and only if the following is true for all $n_1 \in \mathbb{A}$, and for all encryptions $t = \{\!|t_0|\!\}_K$ sets of encryptions S. If t occurs outside S in any $n_1 \in \mathbb{A}$, then either (1) K is potentially compromised before n_1 in $\mathbb{A}$, or (2) for some K_0 with $\{\!|t|\!\}_{K_0} \in S$, K_0^{-1} is potentially compromised before n_1 in $\mathbb{A}$, or (3) t exits S at some positive regular $m_1 \preceq_{\mathbb{A}} n_1$.

We say that $\mathbb{A}$ *respects origination* if $n \preceq_{\mathbb{A}} m$ whenever, for any $a \in \mathsf{unique}_{\mathbb{A}}$, a originates at $n \in \mathbb{A}$ and a is mentioned in $\mathsf{msg}(m)$.

Proposition 5 (Completeness of Authentication Tests). *Let $\mathbb{A}$ respect origination. $\mathbb{A}$ satisfies the outgoing and incoming authentication tests if and only if there exists a bundle $\mathcal{B}$ such that $\mathbb{A}$ is tightly embedded into $\mathcal{B}$.*

Proof Sketch. From right to left, use Props. 2–3. From left to right, we must show that for every negative $n \in \mathbb{A}$, $\mathsf{msg}(n)$ is potentially compromised before n in $\mathbb{A}$. To do so, for any given negative $n \in \mathbb{A}$, we apply Prop. 4 to:

1. $V = \mathsf{non}_{\mathbb{A}} \cup U$ where $U = \mathsf{unique}_{\mathbb{A}} \cap \{a \colon a \text{ originates in } \mathbb{A}\}$; and
2. $S_{\mathrm{spt}} = \{\mathsf{msg}(m) \colon m \preceq_{\mathbb{A}} n \wedge m \text{ positive }\}$. □

Conclusion. We have presented two principles about how messages are transformed in cryptographic protocols. These two principles are complete in the sense that whenever they are satisfied in a skeleton, then that skeleton describes a possible execution of the protocol, modulo some choice of adversary behavior.

This result is part of the justification for the search method of CPSA, which is based on the authentication tests [5]. CPSA tries to complete partial executions,

by which we mean skeletons that are not tightly embedded into any bundle. It uses the authentication tests to consider what ingredients may need to be added to obtain a minimal execution. By considering the ingredients suggested by the authentication tests, it finds all minimal, essentially different executions. Thus, this paper provides the core justification for the claim in [5] that the search finds all the possibilities.

References

1. Martín Abadi and Bruno Blanchet. Analyzing security protocols with secrecy types and logic programs. *Journal of the ACM*, 52(1):102–146, January 2005.
2. L. Bozga, Y. Lakhnech, and M. Perin. Pattern-based abstraction for verifying secrecy in protocols. In *Tools and Algorithms for Construction and Analysis of Systems (TACAS)*, number 2619 in LNCS. Springer, 2003.
3. Michael Burrows, Martín Abadi, and Roger Needham. A logic of authentication. *Proceedings of the Royal Society*, Series A, 426(1871):233–271, December 1989.
4. Shaddin F. Doghmi, Joshua D. Guttman, and F. Javier Thayer. Completeness of the authentication tests. In J. Biskup and J. Lopez, editors, *European Symposium on Research in Computer Security (ESORICS)*, number 4734 in LNCS, pages 106–121. Springer-Verlag, September 2007.
5. 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)*, number 4424 in LNCS, pages 523–538. Springer, March 2007. Extended version at URL:`http://eprint.iacr.org/2006/435`.
6. Daniel Dolev and Andrew Yao. On the security of public-key protocols. *IEEE Transactions on Information Theory*, 29:198–208, 1983.
7. Nancy Durgin, Patrick Lincoln, John Mitchell, and Andre Scedrov. Multiset rewriting and the complexity of bounded security protocols. *Journal of Computer Security*, 12(2):247–311, 2004. Initial version appeared in *Workshop on Formal Methods and Security Protocols*, 1999.
8. Andrew D. Gordon and Alan Jeffrey. Types and effects for asymmetric cryptographic protocols. *Journal of Computer Security*, 12(3/4):435–484, 2003.
9. 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.
10. Joshua D. Guttman and F. Javier Thayer. Authentication tests. In *Proceedings, 2000 IEEE Symposium on Security and Privacy*. May, IEEE Computer Society Press, 2000.
11. Joshua D. Guttman and F. Javier Thayer. Authentication tests and the structure of bundles. *Theoretical Computer Science*, 283(2):333–380, June 2002. Conference version appeared in *IEEE Symposium on Security and Privacy*, May 2000.
12. Joshua D. Guttman, F. Javier Thayer, Jay A. Carlson, Jonathan C. Herzog, John D. Ramsdell, and Brian T. Sniffen. Trust management in strand spaces: A rely-guarantee method. In David Schmidt, editor, *Programming Languages and Systems: 13th European Symposium on Programming*, number 2986 in LNCS, pages 325–339. Springer, 2004.
13. Lawrence C. Paulson. The inductive approach to verifying cryptographic protocols. *Journal of Computer Security*, 1998. Also Report 443, Cambridge University Computer Lab.

14. Adrian Perrig and Dawn Xiaodong Song. Looking for diamonds in the desert: Extending automatic protocol generation to three-party authentication and key agreement protocols. In *Proceedings of the 13th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, July 2000.

A Proof of Proposition 4

Proposition. *Let V be a set of atoms; let S_{spt} be a finite set of terms; and let t_1 be a term such that, for any $a \in V$, if $a \sqsubseteq t_1$, then $a \sqsubseteq t_0$ for some $t_0 \in S_{\mathrm{spt}}$. Suppose the following conditions hold:*

1. *for all $a \in V$ and all sets of encryptions S_{enc}, if a has exited S_{enc} passing from S_{spt} to t_1, then there is some $\{|t|\}_{K_0} \in S_{\mathrm{enc}}$, such that ${K_0}^{-1}$ is derivable from S_{spt} avoiding V; and*
2. *for all encryptions $\{|t|\}_K$, and all sets of encryptions S_{enc}, if $\{|t|\}_K$ has exited S_{enc} passing from S_{spt} to t_1, then either K is derivable from S_{spt} avoiding V, or else some ${K_0}^{-1}$ with $K_0 \in \mathsf{used}(S_{\mathrm{enc}})$ is derivable from S_{spt} avoiding V.*

Then term t_1 is derivable from S_{spt} avoiding V.

Proof. The proof is by structural induction on the pair (S_{spt}, t_1), i.e. the ordering under which $(S_{\mathrm{spt}}, t_1) \leq (S'_{\mathrm{spt}}, t'_1)$ iff $t_1 \sqsubseteq t'_1$, and for all $t \in S_{\mathrm{spt}}$, there is some $t' \in S'$ such that $t \sqsubseteq t'$.

Case $t_1 = a$**:** If $a \notin V$, then the one-node web originating a satisfies the conditions. Otherwise, $S^a = \{t \in S_{\mathrm{spt}} \colon a \sqsubseteq t\}$ is non-empty. If $a \in S^a$, then the empty web suffices. If some concatenation $t_0 \mathbin{\hat{}} t'_0 \in S^a$, then apply the induction hypothesis to $S^a \setminus \{t_0 \mathbin{\hat{}} t'_0\} \cup \{t_0\} \cup \{t'_0\}$. This asserts the existence of a penetrator web G_a deriving a. Obtain the desired web by prepending a separation S-strand above any occurrences of t_0 and t'_0 in G_a.
Otherwise, S^a consists entirely of encryptions, and a has exited S^a passing from S_{spt} to a. By condition 1, there is some $\{|t|\}_{K_0} \in S^a$ with ${K_0}^{-1}$ derivable from S_{spt} avoiding V, using some web $G_{{K_0}^{-1}}$. Thus, applying the induction hypothesis to $(S^a \setminus \{\{|t|\}_{K_0}\}) \cup \{t\}$, we obtain a web G. We may prepend $G_{{K_0}^{-1}}$ and a decryption D-strand before G to obtain the required web.

Case $t_1 = t'_1 \mathbin{\hat{}} t''_1$**:** Apply the induction hypothesis to t'_1 and t''_1, and append a concatenation C-strand after the resulting webs.

Case $t_1 = \{|t'_1|\}_K$**:** Suppose K is derivable from S_{spt} avoiding V, using some web G_K. Apply the induction hypothesis to t'_1, obtaining a web G. Append an encryption E-strand after G_K and G to derive $\{|t'_1|\}_K$.
Otherwise, by condition 2, some K_0^{-1} with $\{|t_0|\}_{K_0} \in S_{\mathrm{enc}}$ is derivable from S_{spt} avoiding V, using a web $G_{{K_0}^{-1}}$. Apply the induction hypothesis to $(S_{\mathrm{spt}} \setminus \{\{|t_0|\}_{K_0}\}) \cup \{t_0\}$, obtaining a web G. Prepend $G_{{K_0}^{-1}}$ and a decryption D-strand before G. □