Authentication Tests and the Normal Penetrator*

Joshua D. Guttman F. Javier Thayer Fábrega

The MITRE Corporation
{guttman, jt}@mitre.org

February, 2000

Abstract

Suppose a principal in a cryptographic protocol creates and transmits a message containing a new value v, later receiving v back in a different cryptographic context. It can conclude that some principal possessing the relevant key has received and transformed the message emitting v. In some circumstances, this principal must be a regular participant of the protocol, not the penetrator.

An inference of this kind is an *authentication test*. We introduce two main kinds of authentication test. An outgoing test is one in which the new value v is transmitted in encrypted form, and only a regular participant can extract it from that form. An incoming test is one in which v is received back in encrypted form, and only a regular participant can put it in that form. We combine these two tests with a supplementary idea, the unsolicited test, and a related method for checking that certain values remain secret. Together, these techniques determine what authentication properties are achieved by a wide range of cryptographic protocols.

In this paper we introduce authentication tests and prove their soundness. We illustrate their power by giving new and straightforward proofs of security goals for several protocols. We also illustrate how to use the authentication tests as a heuristic for finding attacks against incorrect protocols. Finally, we suggest a protocol design process.

We express these ideas in the strand space formalism [24], which provides a convenient context to prove them correct.

*This work was supported by the National Security Agency through US Army CECOM contract DAAB07-99-C-C201. A shorter version of this paper (without proofs) appeared as [8]. The present version is identical to MITRE Technical Report MTR 00B04.

Contents

List of Figures

1 Introduction

One reason why cryptographic protocol analysis is hard is that the attacker has so many choices. He may apply a repertory of actions in any order to any message he observes, and he may submit the results in place of any legitimate message. In addition, the attacker may initiate new sessions of the protocol, or await sessions initiated by regular participants [6]. Consequently, even though cryptographic protocols are simple finite state activities in the absence of an attacker, the analysis of possible attacks is not necessarily decidable; indeed, even if the protocols are restricted so that the problem is decidable, it may not be tractable [2].

In this paper we use the strand space formalism [24] to restrict the order in which the penetrator applies the operations available to him (Section 3). Anything the penetrator can do at all, he can do carrying out operations in this restricted order. There are two ingredients in the restriction, a normal form lemma (Section 3.2, Proposition 3.8), and an efficiency condition (Section 3.6, Proposition 3.22). The normal form lemma is not new [4, 2], although the efficiency condition appears to be.

The main novelty in this paper are some very simple-to-apply methods for authentication and secrecy results, which the penetrator restrictions justify. An important consequence of the restrictions is that, for certain encrypted components of messages, the penetrator cannot take any significant action. Those components may be discarded, but if they are delivered to a regular participant, they can only be delivered unaltered. Only the regular participants can change these encrypted components in the way demanded by the protocol.

Therefore this kind of component may be regarded as an *authentication test component*: if the contents are later received in transformed form, then only a regular participant, not the penetrator, can have transformed them. In favorable circumstances, it can only be one regular participant, the intended one, who has thereby been authenticated.

We embody these ideas in three authentication results (Section 4.2, Authentication Tests 1–3). These results allow us to establish many authentication results without any consideration of the dynamic execution of protocols, involving the activity of several principals. Instead, it suffices to consider the forms of the possible behaviors of the principals independently. We use the Needham-Schroeder-Lowe protocol [16, 12] in explaining the ideas. In Section 5, we illustrate the authentication tests by proving the authentication properties of some familiar protocols and identifying counter-examples to others. The protocols we consider are Otway-Rees [18], Neuman-Stubblebine [17], and Woo-Lam [25, 26]. It is routine to apply the method to new protocols, whether they use public keys or shared symmetric keys.

However, not every protocol can be verified using these methods. In particular, for the authentication theorems to apply, the protocol must not allow the authentications tests to be proper sub-messages of other messages manipulated by the regular participants. We end (Section 6) by suggesting it as a design principle that protocols avoid this sort of nesting, and concentrate the

crucial parameters to be authenticated in a small number of authentication test components.

The authentication tests themselves are easy to apply, but the proofs justifying them are more complicated. We would compare the authentication tests to the interface to a module; the implementation internal to the module is complex, but the interface makes it easy to use its services without worrying about the internals. For some purposes it would be helpful to enlarge the interface. There are additional services, or ways of drawing conclusions about authentication protocols, that the proof methods of Sections 3 and 4 can offer. For instance, one addition would be to make explicit the order in which events have occurred, which gives a convenient way to reason about whether a key has been generated recently. An enrichment of the message algebra would explicitly model the way a key may be generated by hashing other values (as is used e.g. in the SSL and TLS protocols [5]). However, the authentication tests currently exported in Section 4 already apply to a wide range of protocols, and give a highly intuitive explanation for why they are right, or where they go wrong.

The proof methods of Section 3 can be used for other purposes also; in [9] we use them to study when different protocols may be safely combined.

1.1 Strand Spaces

We very briefly summarize the ideas behind the strand space model [24]; see also Appendix A.

A is the set of messages that can be sent between principals. We call elements of A *terms*. A is freely generated from two disjoint sets, T (representing texts such as nonces or names) and K (representing keys) by means of concatenation and encryption. The concatenation of terms g and h is denoted $g\,h$, and the encryption of h using key K is denoted $\{|h|\}_K$. (See Appendix A.3.)

For example, in the Needham-Schroeder protocol [16], the initiator A sends a term of the form $\{|N_a\,A|\}_{K_B}$ to start an exchange intended for B. This is a ciphertext created using B's public key K_B; the plaintext is the result of concatenating a nonce (random bitstring) N_a and A's name.

A term t is a *subterm* of another term t', written $t \sqsubset t'$, if starting with t we can reach t' by repeatedly concatenating with arbitrary terms and encrypting with arbitrary keys. Hence, $K \not\sqsubset \{|t|\}_K$, except in case $K \sqsubset t$. The subterms of t are the values that are uttered when t is sent; in $\{|t|\}_K$, K is not uttered but used. (See Definition A.7.)

For instance, the subterms of $\{|N_a\,A|\}_{K_B}$ are N_a, A, the concatenated message $N_a\,A$, and $\{|N_a\,A|\}_{K_B}$ itself. The key K_B is not part of what is uttered, it just contributes to *how* the message is constructed.

A *strand* is a sequence of message transmissions and receptions, where transmission of a term t is represented as $+t$ and reception of term t is represented as $-t$. A strand element is called a *node*. If s is a strand, $\langle s, i\rangle$ is the i^{th} node on s. The relation $n \Rightarrow n'$ holds between nodes n and n' if $n = \langle s, i\rangle$ and $n' = \langle s, i+1\rangle$. Hence, $n \Rightarrow^+ n'$ means that $n = \langle s, i\rangle$ and $n' = \langle s, j\rangle$ for some

$j > i$. The relation $n \rightarrow n'$ represents inter-strand communication; it means that $\mathrm{term}(n_1) = +t$ and node $\mathrm{term}(n_2) = -t$.

Continuing with the Needham-Schroeder protocol as our pedagogical illustration, an initiator strand offers a sequence of events of the form

$$\langle +\{\!|N_a\, A|\!\}_{K_B},\quad -\{\!|N_a\, N_b|\!\}_{K_A},\quad +\{\!|N_b|\!\}_{K_B}\rangle$$

In this strand s_i, the initiator A sends a term $\{\!|N_a\, A|\!\}_{K_B}$ intended for the responder B, and expects to receive back a term of the form $\{\!|N_a\, N_b|\!\}_{K_A}$, after which it will send $\{\!|N_b|\!\}_{K_B}$. The reception is $\langle s_i, 2\rangle$ and the final transmission is $\langle s_i, 3\rangle$. The responder strands offer a sequence of events of the complementary form

$$\langle -\{\!|N_a\, A|\!\}_{K_B},\quad +\{\!|N_a\, N_b|\!\}_{K_A},\quad -\{\!|N_b|\!\}_{K_B}\rangle$$

When the data values match N_a, A, ..., match between an initiator strand s_i and a responder strand s_r, then we have $\langle s_i, 1\rangle \rightarrow \langle s_r, 1\rangle$ and $\langle s_r, 2\rangle \rightarrow \langle s_i, 2\rangle$. An initiator or responder strand has four parameters (or degrees of freedom), namely the two nonces N_a and N_b and the two names A and B. For this illustration, we regard the public keys K_A and K_B to be reliably determined from A and B, possibly by some public key infrastructure. When we write $s_i \in \mathrm{NSInit}[A, B, N_a, N_b]$ in this illustration, we will mean that s_i is an initiator strand using the particular values shown as parameters, and similarly for $s_r \in \mathrm{NSResp}[A, B, N_a, N_b]$. The principal active in $\mathrm{NSInit}[A, B, N_a, N_b]$ as initiator is A, while the principal active in $\mathrm{NSResp}[A, B, N_a, N_b]$ as responder is A.

A *strand space* Σ is a set of strands. Σ typically will not contain strands of every possible kind $\mathrm{NSInit}[A, B, N_a, N_b]$ and $\mathrm{NSResp}[A, B, N_a, N_b]$, modeling the fact that nonces are chosen from a large set and are used very sparsely, even over substantial periods. The two relations $\Rightarrow$ and $\rightarrow$ jointly impose a graph structure on the nodes of Σ. The vertices of this graph are the nodes, and the edges are the union of $\Rightarrow$ and $\rightarrow$.

We say that a term t *originates* at a node $n = \langle s, i\rangle$ if the sign of n is positive; $t \sqsubset \mathrm{term}(n)$; and $t \not\sqsubset \mathrm{term}(\langle s, i'\rangle)$ for every $i' < i$. Thus, n represents a message transmission that includes t, and it is the first node in s including t. For instance, if $s_i \in \mathrm{NSInit}[A, B, N_a, N_b]$, then N_a and A both originate at $\langle s_i, 1\rangle$. If $s_r \in \mathrm{NSResp}[A, B, N_a, N_b]$, then N_b originates at $\langle s_r, 2\rangle$, assuming that N_b is distinct from N_a and A, which have already been received at $\langle s_r, 1\rangle$.

If a value originates on only one node in the strand space, we call it *uniquely originating*; uniquely originating values are desirable as nonces and session keys. In a particular strand space, a nonce N_a may originate uniquely on $\langle s_i, 1\rangle$, in which case there is at most one strand in $\mathrm{NSInit}[A, B, N_a, N_b]$. A is unlikely to originate uniquely, because the same name will be used in many runs with many partners. When we assume that a value like N_a originates uniquely in some strand space Σ, we are effectively assuming that Σ is not unrealistically large, so large as to have independent events in which the same value is repeatedly chosen at random from a large set.

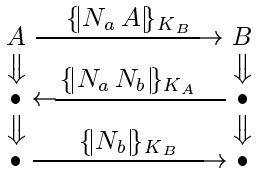


Figure 1: A Bundle: Intended Run of Needham-Schroeder

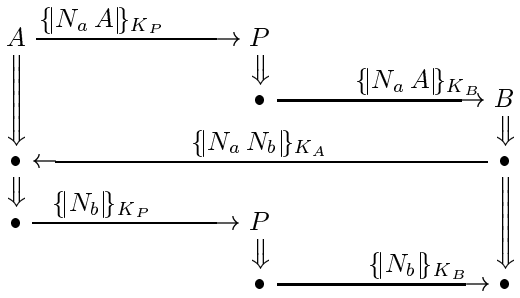


Figure 2: A Bundle: Penetrated Run of Needham-Schroeder

A bundle is a causally well-founded collection of nodes and arrows of both kinds. In a bundle, when a strand receives a message m, there is a unique node transmitting m from which the message was immediately received. By contrast, when a strand transmits a message m, many strands (or none) may immediately receive m. (See Definition A.3.) The height of a strand in a bundle is the number of nodes on the strand that are in the bundle. Authentication theorems generally assert that a strand has at least a given height in some bundle, meaning that the principal must have engaged in at least that many steps of its run. Two illustrative bundles are shown in Figures 1–2. In Figure 1, initiator and responder match strands in the expected way, while in Figure 2 a penetrator manipulates B into believing that A is having a session with B, whereas in fact A intends to have a session with P [11, 12]. More formally, the strand on the left side is in the set $\mathrm{NSInit}[A, P, N_a\ N_b]$, not $\mathrm{NSInit}[A, B, N_a, N_b]$.

Given any bundle $\mathcal{C}$, there is a natural partial ordering on the nodes of $\mathcal{C}$, which we refer to as $\preceq_{\mathcal{C}}$, according to which $n_1 \preceq_{\mathcal{C}} n_2$ if there is a path from n_1 to n_2 using zero or more arrows of either kind (Definition A.5). This relation expresses the fact that n_1 causally contributes to n_2 occurring in $\mathcal{C}$. In Figures 1 and 2, the relation happens to be a linear ordering, but this is not true in Figure 3, where neither K node is accessible from the other.

A strand represents the local view of a participant in a run of a protocol. For a legitimate participant, it represents the messages that participant would send or receive as part of one particular run of his side of the protocol. We call

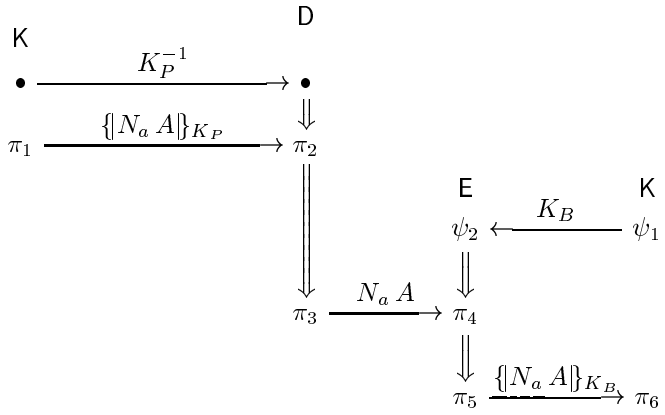


Figure 3: Penetrator Strands for Needham-Schroeder Attack

a strand representing a legitimate participant a *regular* strand. For the penetrator, the strand represents an atomic deduction. More complex actions can be formed by connecting several penetrator strands. While regular principals are represented only by what they say and hear, the behavior of the penetrator is represented more explicitly, because the values he deduces are treated as if they had been said publicly.

We partition penetrator strands according to the operations they exemplify. E-strands encrypt when given a key and a plaintext; D-strands decrypt when given a decryption key and matching ciphertext; C-strands and S-strands concatenate and separate terms, respectively; K-strands emit keys from a set of known keys; and M-strands emit known atomic texts or guesses. (See Definition A.9.)

As an example, the compound behavior of the penetrator P, shown at the center top in Figure 2, can be realized using several of our official penetrator strands as shown in Figure 3, in which nodes π_1 and π_6 represent the nodes shared with Figure 2. This figure should be regarded as a part of Figure 2, shown separately simply to reduce its complexity. Some nodes have been labelled for later use.

In Figure 3, the penetrator emits a private key K_P^{-1} that is known to himself, and uses the result on a D strand to decrypt the incoming message. He emits a public key K_B known (presumably) to everyone, using it in an encryption strand to produce the term $\{|N_a\, A|\}_{K_B}$, needed to start the process of duping B. The other penetrator action shown in Figure 2 may be expanded in a similar manner.

1.2 New Components

When a node transmits or receives a concatenated message, the penetrator—using C-strands and S-strands—has full power over how the parts are concatenated together. Thus, the important units for protocol correctness are what we call the *components*. A term t_0 is a component of t if $t_0 \sqsubset t$, t_0 is not a concatenated term, and every $t_1 \neq t_0$ such that $t_0 \sqsubset t_1 \sqsubset t$ is a concatenated term. Components are either atomic values or encryptions. (See Definition A.8.) For instance, the term $\{|N_a\, A|\}_{K_B}$ consists of a single component, while $N_a\, A$ has two components, the atomic values N_a and A. We say t is a component of a node n if t is a component of term(n).

A term t is *new* at $n = \langle s, i\rangle$ if t is a component of term(n), but t is not a component of node $\langle s, j\rangle$ for every $j < i$ (Definition A.8). A component is new even if it has occurred earlier as a nested subterm of some larger component $\cdots \{|\cdots t \cdots|\}_K \cdots$. For instance, $\{|N_a\, A|\}_{K_P}$ is new on the top left node of Figure 2, and N_a is new on the last node of the D strand in Figure 3.

When a component occurs new on a regular node, then the principal executing that strand has done some cryptographic work to produce the new component. The idea of emphasizing components and the regular nodes at which they occur new is due to Song [21].

2 Bundle Equivalences and Graph Operations

2.1 Bundle Equivalence

Definition 2.1 *Bundles $\mathcal{C}, \mathcal{C}'$ on a strand space Σ are equivalent iff they have the same regular nodes.*

A set ϕ of bundles is invariant under bundle equivalences *if whenever bundles $\mathcal{C}$ and $\mathcal{C}'$ are equivalent, $\mathcal{C} \in \phi \Rightarrow \mathcal{C}' \in \phi$.*

Agreement and non-injective agreement properties [14, 24, 27] are invariant under bundle equivalences in this sense. For instance, a non-injective agreement property, expressed in our framework, asserts that whenever a bundle contains a protocol strand (for instance, a responder strand) of a certain height, then it also contains a matching strand (for instance, an initiator strand using the same data values) of suitable height. As such, it always concerns what nodes, representing regular activity of the protocol, must be present in bundles. Penetrator activity may or may not be present.

Secrecy properties may also be expressed in a form that is invariant under bundle equivalences. We say (temporarily) that a value t is uncompromised in $\mathcal{C}$ if for every $\mathcal{C}'$ equivalent to $\mathcal{C}$, there is no node $n \in \mathcal{C}'$ such that term$(n) = t$. In this form, a value is uncompromised if the penetrator cannot extract it in explicit form without further cooperation of regular strands. When stated in this form, the assertion that a value is uncompromised is invariant under bundle equivalences.

2.2 Graph Operations

A graph operation on a bundle $\mathcal{C}$ consists of a sequence of one or more of the following:

1. Deletion of any set of penetrator strands from the bundle.

2. Addition of edges $n \to n'$ with $\mathrm{term}(n) = +a$, $\mathrm{term}(n') = -a$.

3. Deletion of edges $n \to n'$.

A graph operation yields a new graph $\mathcal{C}'$. However, the graph $\mathcal{C}'$ is not necessarily a bundle. For instance, if $n \to n'$ is an edge of $\mathcal{C}$ with n a penetrator node, removal of the strand that contains n is a graph operation which causes the resulting graph to have a negative node with no in-arrow.

A *lonely node* in a strand space graph is a node with no incoming edge (if the node is negative) or no outgoing edge (if the node is positive). Lonely negative nodes are ruled out by the definition of bundle, whereas lonely positive nodes are allowed. Similarly call a node in a strand space graph *gregarious* if it has more than one edge leaving or entering it. Gregarious negative nodes are ruled out, whereas gregarious positive ones are allowed. In applying graph operations on bundles, we must be careful not to create lonely or gregarious negative nodes.

Proposition 2.2 *Suppose $\mathcal{C}$ is a bundle and $\mathcal{C}'$ is obtained from $\mathcal{C}$ by a graph operation such that*

1. *For any edge $n \mapsto n'$ of $\mathcal{C}'$ there is a sequence of nodes and bundle edges $n = n_1 \mapsto \cdots \mapsto n_k = n'$ in $\mathcal{C}$.*

2. *$\mathcal{C}'$ has no lonely or gregarious negative nodes.*

Then $\mathcal{C}'$ is a bundle. Moreover, $\mathcal{C}'$ is equivalent to $\mathcal{C}$ and the ordering on $\mathcal{C}'$ is a restriction of the ordering on $\mathcal{C}$.

Proof. The nodes in any connected sequence in $\mathcal{C}'$ is a subsequence of the nodes of a connected sequence in $\mathcal{C}$. To show $\mathcal{C}'$ is acyclic, notice that by assumption 1, for any non-trivial cycle in $\mathcal{C}'$ there is a non-trivial cycle in $\mathcal{C}$. Thus $\mathcal{C}'$ is a bundle. It is equivalent to $\mathcal{C}$ because a graph operation modifies only the set of penetrator nodes included in the bundle. ■

3 Redundancies and Paths

We turn our attention to the portions of a bundle that contain penetrator activity, and the ways that we can simplify those portions.

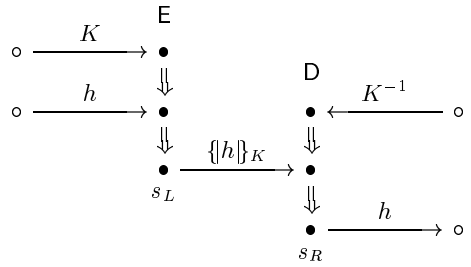


Figure 4: E-D Redundancy

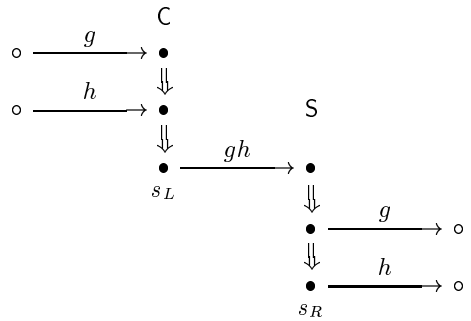


Figure 5: C-S Redundancy

3.1 Redundancies

Definition 3.1 *A* redundancy *in a bundle $\mathcal{C}$ is any labeled subgraph of $\mathcal{C}$ that has one of the forms given in Figures 4-5.*

Each redundancy contains nodes on two penetrator strands, indicated by the symbol $\bullet$, and a number of "fringe" nodes indicated by the symbol $\circ$. The nodes are connected by inward edges $\circ \to \bullet$, outward edges $\bullet \to \circ$ and internal edges $\bullet \to \bullet$. The fringe nodes $\circ$ may be either regular nodes or penetrator nodes.

The presence of redundancies in a bundle makes it more difficult to see what the penetrator can actually do, and in particular whether any attacks can be crafted by a circuitous combination of strands. The purpose of this section is to show redundancies can be eliminated without any weakening of the penetrator's capability.

Proposition 3.2 *Given any bundle $\mathcal{C}$ there exists an equivalent bundle $\mathcal{C}'$ with no redundancies. Moreover, the penetrator nodes of $\mathcal{C}'$ is a subset of the penetrator nodes of $\mathcal{C}$ and the ordering $\prec_{\mathcal{C}'}$ is a restriction of the ordering $\prec_{\mathcal{C}}$. If there exists $n \in \mathcal{C}$ such that $term(n) = t$, then there exists $n' \in \mathcal{C}'$ such that $term(n') = t$.*

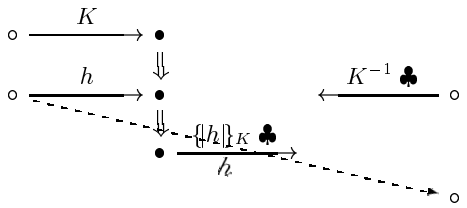


Figure 6: E-D Redundancy Elimination

PROOF. Consider each one of the redundancy types shown in figures 4-5. Each one of these redundancies is a subgraph of $\mathcal{C}$ consisting of two penetrator strands s_L and s_R, some arrows into the subgraph and some arrows out of the subgraph. Notice that by suitably replicating the strand s_R in each one of the redundancy cases, we can assume the positive nodes of s_R are not gregarious, that is, have exactly one outgoing arrow. For each such subgraph,

1. Add the edges indicated by the dotted lines as shown in figures 6-7. In the case of C-S elimination, two new edges are added; in the case of E-D elimination only one new edge is added. For each such new edge $n \to n'$, there is clearly a path $n \mapsto n_1 \mapsto \cdots \mapsto n_k = n'$ in $\mathcal{C}$. Note that the addition of this edge creates some gregarious positive and negative nodes. In the next step we will remove the redundant edges leading to the gregarious negative nodes.

2. Delete the right penetrator strand s_R. As a result of removing s_R, those edges $m \to n'$ going out of s_R are removed as well. In step 1, we added an arrow into n' so that removal of $m \to n'$ does not leave us with a lonely negative node.

3. As a result of the previous step, some positive nodes may have no outgoing arrows. These are shown by ♣ in the figure. However, the presence of lonely positive nodes does not violate the bundle property so no further action is necessary to deal with these.

Note that the graph operation above satisfies the conditions of Proposition 2.2. Hence, the resulting graph is bundle equivalent to $\mathcal{C}$, and its ordering is a restriction of the ordering of $\mathcal{C}$. Note also that for each of the deleted nodes on s_R, there is another node with the same term that does not lie on s_R. ■

3.2 Penetrator Paths and Normal Bundles

$m \Rightarrow^+ n$ means that m, n are nodes on the same strand with n occuring after m (Definition A.2, Clause 4). The notation $m \longmapsto n$ means:

- either $m \Rightarrow^+ n$ with term(m) negative and term(n) positive, or else

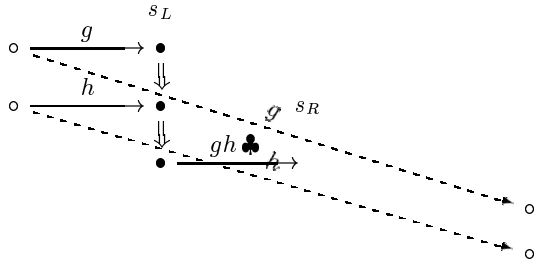


Figure 7: C-S Redundancy Elimination

- $m \to n$.

A *path* p through $\mathcal{C}$ is any finite sequence of nodes and edges $n_1 \longmapsto n_2 \longmapsto \cdots \longmapsto n_k$. Clearly, $n \prec_{\mathcal{C}} n'$ whenever there is a path $n = n_1 \longmapsto n_2 \longmapsto \cdots \longmapsto n_k = n'$. We assume all paths begin on a positive node, and end on a negative node.

We refer to the ith node of the path p as p_i. The length of p will be $|p|$, and we will write $\ell(p)$ to mean $p_{|p|}$, i.e. the last node in p.

A penetrator path is one in which all nodes other than possibly the first or the last node are penetrator nodes. As an example of a penetrator path, in which the first and last nodes are in fact regular, consider again the partial bundle shown in Figure 3. The path $\pi =$

$$\pi_1 \to \pi_2 \Rightarrow^+ \pi_3 \to \pi_4 \Rightarrow^+ \pi_5 \to \pi_6$$

is a path that traverses penetrator nodes, connecting A's first transmission $\{|N_a\, A|\}_{K_P}$ to B's first reception $\{|N_a\, A|\}_{K_B}$. By contrast, the path $\psi =$

$$\psi_1 \to \psi_2 \Rightarrow^+ \pi_5 \to \pi_6$$

starts on a penetrator node and ends on a regular node.

Definition 3.3 *Given a path p, one $\Rightarrow^+$ edge immediately precedes another $\Rightarrow^+$ edge in p iff they are separated in p by a single $\to$ edge.*

For instance, $\pi_2 \Rightarrow^+ \pi_3$ immediately precedes $\pi_4 \Rightarrow^+ \pi_5$ in π.

Consider a $\Rightarrow^+$-edge between penetrator nodes. There are four penetrator strand types with a negative node followed by a positive node, namely E, D, C, and S strands.

Definition 3.4 *A $\Rightarrow^+$-edge is* constructive *if it is part of a E or C strand. It is* destructive *if it is part of a D or if it is part of a S strand.*

A penetrator node n is initial *if it is a K or M node.*

Any penetrator path that begins at a regular node contains only constructive and destructive $\Rightarrow^+$-edges, because initial nodes can occur only at the beginning of a path.

Proposition 3.5 *In a bundle, a constructive edge immediately followed by a destructive edge has one of the following two forms:*

1. *Part of a $\mathsf{E}_{h,K}$ immediately followed by part of a $\mathsf{D}_{h,K}$ strand for some h, K*
2. *Part of a $\mathsf{C}_{g,h}$ immediately followed by part of a $\mathsf{S}_{g,h}$ strand for some g, h.*

PROOF. This follows immediately from freeness of the message algebra.

Proposition 3.6 *If the bundle $\mathcal{C}$ has no redundancies of type C-S and E-D, then for any penetrator path of $\mathcal{C}$, every destructive edge precedes every constructive edge.*

PROOF. If some constructive edge precedes a destructive one, then some constructive edge immediately precedes a destructive one. However, if the bundle has no redundancies, then by Proposition 3.5, a constructive edge cannot immediately precede a destructive one. ■

Since the property just introduced is very important, we give it a name, stressing the analogy with Prawitz's notion of normal derivation [20]:

Definition 3.7 *A bundle $\mathcal{C}$ is* normal *if, for any penetrator path of $\mathcal{C}$, every destructive edge precedes every constructive edge.*

Clarke et al. [4] first observed the analogy between penetrator activities and natural deduction inferences. By Propositions 3.2 and 3.6, we may infer:

Proposition 3.8 (Penetrator Normal Form Lemma) *For any bundle $\mathcal{C}$ there exists an equivalent normal bundle $\mathcal{C}'$.*

Moreover, the penetrator nodes of $\mathcal{C}'$ form a subset of the penetrator nodes of $\mathcal{C}$ and the ordering $\prec_{\mathcal{C}'}$ is a restriction of the ordering $\prec_{\mathcal{C}}$. If there exists $n \in \mathcal{C}$ such that $term(n) = t$, then there exists $n' \in \mathcal{C}'$ such that $term(n') = t$.

3.3 Rising and Falling Paths

Definition 3.9 *A penetrator path is* falling *if for all adjacent nodes $n \longmapsto n'$ on the path $term(n') \sqsubset term(n)$.*

A penetrator path is rising *if for all adjacent nodes $n \longmapsto n'$ on the path $term(n) \sqsubset term(n')$.*

The path π from Figure 3 contains a falling subpath $\pi_1 \longmapsto \cdots \longmapsto \pi_4$ and a rising subpath $\pi_3 \longmapsto \cdots \longmapsto \pi_6$.

A path containing only destructive edges may not be falling, since a destructive path may traverse a decryption strand entering through the key transmission edge (Figure 8). Call the edge labeled K^{-1} in Figure 8 a D-key edge. The other incoming edge into a D strand is a D-cyphertext edge.

In a symmetrical way, a constructive path may traverse an encryption strand entering through the key transmission edge (Figure 9). Call the edge labeled

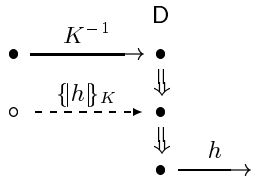


Figure 8: Entering a D strand through a key edge

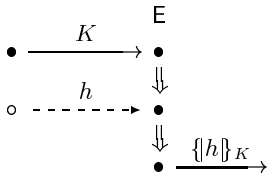


Figure 9: Entering a E strand through a key edge

K in Figure 9 a E-key edge. The other incoming edge into an E strand is an E-plaintext edge. However, in this case we are entitled to a stronger conclusion, because a constructive p can traverse a E-key edge only once, along the edge $p_1 \to p_2$, and only if $\mathrm{term}(p_1) \in \mathsf{K}$. After that we have a compound term, not an atomic key.

The path π from Figure 3 traverses no key edges, while path ψ traverses an E-key edge.

Proposition 3.10 *A destructive path that enters decryption strands only through D-cyphertext edges is falling.*

A constructive path that enters encryption strands only through E-plaintext edges is rising, and this is the case for any constructive p such that $\mathrm{term}(p_1) \notin \mathsf{K}$.

Moreover, the sequence of penetrator strands traversed on a falling path is constrained by the structure of $\mathrm{term}(p_1)$. We use the relation $t_0 \sqsubset_{\mathfrak{K}} t$, which means that t_0 occurs somewhere in t such that every surrounding encryption uses a key $K \in \mathfrak{K}$ (Definition A.8).

Proposition 3.11 *Suppose that p is a falling penetrator path; suppose p_i is a negative penetrator node; and suppose $1 < i < |p|$. Then $\mathrm{term}(p_i)$ is either an encryption or a concatenation, and:*

1. *If $\mathrm{term}(p_i) = \{|h|\}_K$, then p_i lies on a D-strand, and $\mathrm{term}(p_{i+1}) = h$; and*
2. *If $\mathrm{term}(p_i) = g\, h$, then p_i lies on a S-strand, and either $\mathrm{term}(p_{i+1}) = g$ or $\mathrm{term}(p_{i+1}) = h$.*

If p_i is a positive node with $1 \leq i < |p|$, then $\mathrm{term}(p_i) = \mathrm{term}(p_{i+1})$.

Suppose p is a falling penetrator path, and for every D*-strand s that p traverses, with key edge K^{-1}, where $K \in \mathfrak{K}$. Then $term(\ell(p)) \sqsubset_{\mathfrak{K}} term(p_1)$.*

PROOF. The assertion for a positive node p_i is immediate from the definition of paths. So consider a negative node p_i.

Since $i < |p|$, there is a node p_{i+1} on this penetrator path, so p_i is a penetrator node. The strand on which p_i lies is neither a K-strand nor an M-strand, as these lack negative nodes. It is neither a C-strand nor an E-strand because p is a falling path. Hence only D-strands and S-strands remain, and the rest follows from the freeness of the message algebra A.

To see that $\mathrm{term}(\ell(p)) \sqsubset_{\mathfrak{K}} \mathrm{term}(p_1)$ when there exists a falling path traversing only D-strands with decryption keys in $\mathfrak{K}^{-1}$, consider the strands in p in reverse order starting at $\ell(p)$ with $\mathrm{term}(\ell(p))$. For each S-strand, perform a concatenation with the term on the other positive node of that strand (i.e. the positive node not belonging to p). For each D-strand, perform an encryption with the inverse of the decryption key on that strand. The resulting term is $\mathrm{term}(p_1)$. ■

Hence, as we traverse a falling penetrator path, we take successive subterms of the term at the start, with each successive strand determined by the topmost operator of the current term. Observe also that if $\mathrm{term}(\ell(p)) = K$, then there must be some i with $1 \leq i \leq |p|$ and $\mathrm{term}(p_i)$ a component of p_1; simply proceed along the path past all (contiguous) S-strands; if this is $\ell(p)$ then K is the component, while otherwise it is some t_0 with $K \sqsubset t_0$.

Symmetrically, the sequence of penetrator strands traversed on a rising path is constrained by the structure of $\mathrm{term}(\ell(p))$, although we will not need this fact.

One curlicue is useful. A bundle may contain a penetrator D-strand s in which a key K is used to decrypt $\{\!|K|\!\}_{K^{-1}}$, thereby obtaining K. Clearly, we may use a graph operation to splice s out of the bundle, connecting the incoming key edge with term K to the outgoing plaintext edge with term K.

Proposition 3.12 *If $\mathcal{C}$ is any bundle, there is an equivalent bundle $\mathcal{C}'$ containing no* D*-strands of the form $-\{\!|K|\!\}_{K^{-1}} \Rightarrow -K \Rightarrow +K$. The resulting bundle $\mathcal{C}'$ is normal if $\mathcal{C}$ is.*

3.4 Bridges and Bridge Terms

Of special interest are the message transmission edges that come after all destructive edges and before all constructive edges in a normal penetrator path. We call them bridges.

Definition 3.13 *A* bridge *in a bundle $\mathcal{C}$ is a message transmission edge $m \to n$ embedded in a subgraph of one the types shown in Figures 10–13.*

If $m \to n$ is a bridge, then its bridge term *is $term(m)$, which equals $term(n)$.*

A bridge is simple *iff its bridge term is simple, that is, is not of the form $g\,h$.*

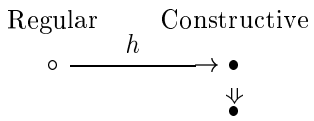


Figure 10: Entry Bridge

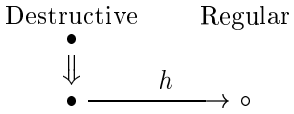


Figure 11: Exit Bridge

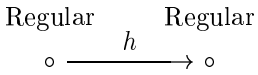


Figure 12: External Bridge

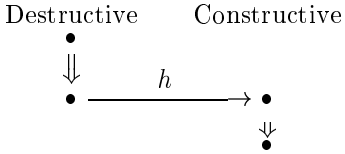


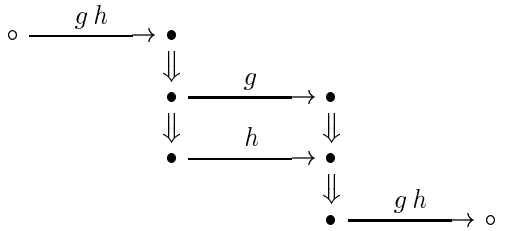
Figure 13: Internal Bridge

Any edge between regular nodes is an external bridge. The source m of a bridge $m \to n$ is never on a constructive penetrator strand, and the target n is never on a destructive penetrator strand. The edge $\pi_3 \to pi_4$ is the only bridge on π.

Proposition 3.14 *Suppose that $\mathcal{C}$ is a normal bundle, and p is any penetrator path in $\mathcal{C}$. Then p traverses exactly one bridge. Any destructive edge along p precedes the bridge of p, and any constructive edge on p follows the bridge of p.*

Any bundle $\mathcal{C}$ can be replaced by an equivalent bundle $\mathcal{C}'$ in which all bridges are simple; moreover if $\mathcal{C}$ is normal so is $\mathcal{C}'$.

PROOF. Consider a bridge $\circ \xrightarrow{g\,h} \circ$ that transmits a concatenated term $g\,h$ from a node on a destructive penetrator node or regular node to a constructive or regular node. Replace the bridge by a graph consisting of two bridges:



These graph operations do not create lonely or gregarious negative nodes and do not introduce cycles in the graph. Moreover if the original bundle is normal, that is contains no C-S or E-D redundancies, the new bundle is also normal. ■

By this proposition, there is a function $\mathrm{pbt}(\cdot)$ from paths to terms that is well-defined on every penetrator path in normal bundles. Given a penetrator path p, $\mathrm{pbt}(p)$ is the *path bridge term* of p, which is the bridge term of the (unique) bridge on p. We may assume that $\mathrm{pbt}(p)$ is always simple, which is to say either an atomic value or an encryption.

The bridge $\pi_3 \to \pi_4$ carries the term $N_a\,A$, so it is not simple. Applying the construction just given in the proof, we obtain two paths; they share their nodes except those bordering the bridges. One path has bridge term N_a, and the other has bridge term A.

A bundle with simple bridges is a kind of worst case scenario, because the penetrator separates and re-concatenates every message between regular nodes. However, simple bridges lead to simple proofs.

3.5 Transforming Edges and Transformation Paths

Our strategy for proving the authentication test results, concerning a test of the form $n \Rightarrow^+ n'$ is to consider the paths leading from n to n'. Because there is a value a originating uniquely at n, and it is received back at n', there must be a path leading from n to n' (apart from the trivial path that follows the strand from n to n'). Moreover, since a is received in a new form at n', there must be a step along the path that changes its form; this is a *transforming edge*.

The incoming and outgoing authentication test results codify conditions under which we can infer that a transforming edge lies on a regular strand. Thus, our proofs focus on the *transformation paths* leading from n to n' that keep track of a "relevant" component containing a. The relevant component changes only when a transforming edge is traversed, and a occurs in a new component.

We regard the edge $n \Rightarrow^+ n'$ as a *transformed edge*, because the same value a occurs in both nodes, but in transformed form.

Definition 3.15 *The edge $n_1 \Rightarrow^+ n_2$ is a* transformed edge *for $a \in \mathsf{A}$ [respectively, a* transforming edge *for $a \in \mathsf{A}$] if n_1 is positive and n_2 is negative [respectively, n_1 is negative and n_2 is positive], $a \sqsubset term(n_1)$, and there is a new component t_2 of n_2 such that $a \sqsubset t_2$.*

Thus, a transformed edge emits a and later tests for its presence in a new form. A transforming edge receives a and later emits it in transformed form. We have chosen to interpret a "form" in which a occurs as a component in which it occurs. Considering again $s_i \in \mathrm{NSInit}[A, B, N_a, N_b]$, the first two nodes

$$+\{\!|N_a\, A|\!\}_{K_B} \;\Rightarrow\; -\{\!|N_a\, N_b|\!\}_{K_A}$$

are a transformed edge for N_a, while the second and third nodes

$$-\{\!|N_a\, N_b|\!\}_{K_A} \;\Rightarrow\; +\{\!|N_b|\!\}_{K_B}$$

are a transforming edge for N_b. Conversely, for $s_r \in \mathrm{NSResp}[A, B, N_a, N_b]$, the first two nodes are a transforming edge for N_a, while the second and third nodes are a transformed edge for N_b.

Definition 3.16 *A* transformation path *is a path for which each node n_i is labelled by a component $\mathcal{L}_i$ of n_i in such a way that $\mathcal{L}_i = \mathcal{L}_{i+1}$ unless $n_i \Rightarrow^+ n_{i+1}$ and $\mathcal{L}_{i+1}$ is new on the strand of n_{i+1}.*

We can regard a transformation path as a sequence of pairs $(n_i, \mathcal{L}_i)$ consisting of a node and a component $\mathcal{L}_i$ of that node. If $\mathcal{L}_i \neq \mathcal{L}_{i+1}$ and $a \sqsubset \mathcal{L}_i$ and $a \sqsubset \mathcal{L}_{i+1}$, then $n_i \Rightarrow^+ n_{i+1}$ is a transforming edge (Definition 3.15) for a. This is the explanation for the name, transformation path. The sequence

$$\begin{array}{ll} \langle & (\pi_1, \{\!|N_a\, A|\!\}_{K_P}),\ (\pi_2, \{\!|N_a\, A|\!\}_{K_P}),\ (\pi_3, N_a), \\ & (\pi_4, N_a),\ (\pi_5, \{\!|N_a\, A|\!\}_{K_B}),\ (\pi_6, \{\!|N_a\, A|\!\}_{K_B}) \quad \rangle \end{array}$$

is a transformation path for N_a. We could also choose a longer example from Figures 2 and 3, because the path p need not be a penetrator path, and need not terminate when a regular node is reached.

By inspecting the forms of penetrator strand (Definition A.9), we observe:

Proposition 3.17 *If $(p, \mathcal{L})$ is a transformation path in which $\mathcal{L}_i \neq \mathcal{L}_{i+1}$, and p_i is a penetrator node, then $p_i \Rightarrow^+ p_{i+1}$ lies either on a D-strand or an E-strand.*

The next proposition states that given a node such as π_6, it is possible to construct a transformation path like the one we have just given, leading back to a node at which N_a originates.

Proposition 3.18 *Suppose $\mathcal{C}$ is a bundle in Σ with $n' \in \mathcal{C}$. If $a \sqsubset t$ where t is a component of n', then there is a transformation path p in $\mathcal{C}$ with $\ell(p) = n'$, $\mathcal{L}_{|p|} = t$, $a \sqsubset \mathcal{L}_i$ for all i and such that a originates at p_1.*

We may choose p so as not to traverse the key edge of a D- *or* E*-strand.*

PROOF. We will construct the path p backwards. Let $n_1 = n'$, let $\mathcal{L}_1 = t$, and suppose that (inductively) we have a transformation path

$$(n_{k+1}, \mathcal{L}_{k+1}) \longmapsto (n_k, \mathcal{L}_k) \longmapsto \cdots \longmapsto (n_1, \mathcal{L}_1)$$

such that $a \sqsubset \mathcal{L}_j$ for all j in the path. If a originates at n_{k+1} then p is complete. So suppose n_{k+1} does not originate at n_{k+1}.

If n_{k+1} is negative, then $\mathcal{C}$ contains a unique n_{k+2} such that $n_{k+2} \rightarrow n_{k+1}$. Extend p backwards to $(n_{k+2}, \mathcal{L}_{k+1})$.

Suppose n_{k+1} is positive. If $\mathcal{L}_{k+1}$ is new, then there exists a node $n_{k+2} \Rightarrow^+ n_{k+1}$ such that $a \sqsubset \mathrm{term}(n_{k+2})$ since a does not originate at n_{k+1}. Extend p backwards to contain some such n_{k+2} and let $\mathcal{L}_{k+2}$ be any component of n_{k+2} which contains a. If $\mathcal{L}_{k+1}$ is not new, then there is a node $n_{k+2} \Rightarrow^+ n_{k+1}$ such that $\mathrm{term}(n_{k+2})$ has a component $\mathcal{L}_{k+1}$. Extend p backwards to $(n_{k+2}, \mathcal{L}_{k+1})$.

Observe that if n_{k+1} is the positive (ciphertext) node on a E-strand, then we may select the plaintext node as n_{k+2}, because it does contain a, and the ciphertext is new. If n_{k+1} is the positive (plaintext) node on a D-strand, then we may select the ciphertext node as n_{k+2}, because it does contain a, and the plaintext is new (by 3.12). Thus, p never traverses a key edge.

Because $\preceq_{\mathcal{C}}$ is a well-founded relation (Proposition A.6) and $i < j$ implies $n_j \prec_{\mathcal{C}} n_i$, eventually $n_j = n$. ■

Proposition 3.19 *Suppose p is a transformation path such that $a \sqsubset \mathcal{L}_i$ for every i and $\mathcal{L}_1 \neq \mathcal{L}_n$. Then p has a transforming edge for a.*

PROOF. Argue by contradiction. If there is no transforming edge for a in the path, then for every edge $(p_i, \mathcal{L}_i) \Rightarrow^+ (p_{i+1}, \mathcal{L}_{i+1})$ in p, there is no new component in p_{i+1} containing a. By definition of transformation path, this means $\mathcal{L}_i = \mathcal{L}_{i+1}$. So in particular, $\mathcal{L}_1 = \mathcal{L}_n$. ■

In the case of our path π, the edges $\pi_2 \Rightarrow p_3$ and $\pi_4 \Rightarrow p_5$ are transforming edges. Note that p_3 lies on a D strand and p_5 lies on a E strand; they are the values p_β and p_α mentioned in the next proposition (respectively).

Proposition 3.20 *Suppose $\mathcal{C}$ be a normal bundle.If $(p, \mathcal{L})$ is a transformation path in $\mathcal{C}$ where p is a penetrator path with $\mathrm{term}(p_1)$ simple, then there is smallest index α such that $\mathrm{term}(p_\alpha) = \mathcal{L}_i = \mathcal{L}_{|p|}$ whenever $|p| \geq i \geq \alpha$. Moreover, if $\mathcal{L}$ is not constant then p_α is the positive node of an* E*-strand.*

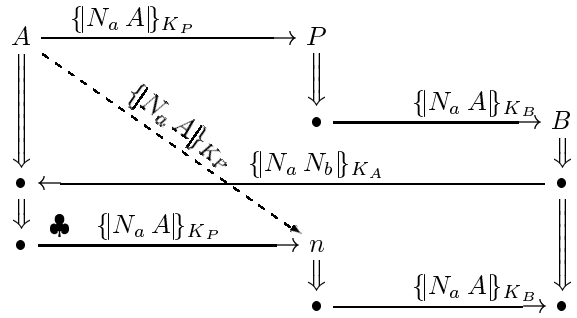


Figure 14: An Inefficient Bundle for a Fictitious Protocol

Similarly, if $(p, \mathcal{L})$ is a transformation path in $\mathcal{C}$ where p is a penetrator path with $\mathrm{term}(\ell(p))$ simple, then either $\mathcal{L}$ is constant or there is a smallest index β such that $\mathcal{L}_\beta \neq \mathcal{L}_1$. p_β is a positive node of a D*-strand and $\mathrm{term}(p_{\beta-1}) = \mathcal{L}_{\beta-1}$.*

In either case, there is always an index β such that $\mathrm{term}(p_\beta) = \mathcal{L}_1$.

Proof. New components of penetrator strands occur only on D-strands or E-strands. Since p is a penetrator path, $\mathcal{L}_{i+1} \neq \mathcal{L}_i$ if and only if p_{i+1} is the positive node of an E-strand or the positive node of a D-strand. If p_{i+1} is the positive node of a E-strand, then $\mathrm{term}(p_{i+1})$ is an encrypted term and therefore $\mathrm{term}(p_{i+1})$ has only one component. Therefore, $\mathrm{term}(p_{i+1}) = \mathcal{L}_{i+1}$. If p_{i+1} is the positive node of a D-strand, then p_i is an encrypted term so that similarly $\mathrm{term}(p_i) = \mathcal{L}_i$.

Notice that if $\mathcal{L}$ is constant and $\mathrm{term}(p_i)$ is simple, then $\mathrm{term}(p_i)$ consists of a single component, and $\mathcal{L}_i = \mathrm{term}(p_i)$. Hence, $\mathcal{L}_1 = \mathcal{L}_{|p|} = \mathrm{term}(p_i)$. ■

3.6 Efficient Bundles

Definition 3.21 *A bundle is* efficient *if and only if, for every node m and negative penetrator node n, if every component of n is a component of m, then there is no regular node m' such that $m \prec m' \prec n$.*

We call a bundle of this kind efficient because the penetrator does the most with what he has rather than making use of additional regular nodes.

The bundles we show in Figure 1 and Figures 2–3 are efficient. Whenever the penetrator node handles a term, there is no earlier node that has all the same components, and a regular node has been traversed in between. However, in the case of the nonsensical variant of the Needham-Schroeder protocol shown in Figure 14, the edge marked ♣ would need to be removed, and replaced with the dashed diagonal. The negative penetrator node n must not receive its term from the third initiator node, when it can be obtained directly from the first initiator node.

Proposition 3.22 *Any bundle $\mathcal{C}$ is equivalent to an efficient bundle $\mathcal{C}'$. $\mathcal{C}'$ may be chosen such that $n \in \mathcal{C}$ implies $n \in \mathcal{C}'$. If a bundle is efficient, then it has an equivalent normal bundle which is also efficient.*

PROOF. Consider a negative penetrator node n and a node m such that every component of n is a component of m. For each component t_0 of m, add an arrow $m \rightarrow$ into a cluster S_{t_0} of S strands to extract the term t_0. This is possible since $t_0 \sqsubset_\emptyset \mathrm{term}(m)$. We refer to the positive S node whose term is t_0 as m'_{t_0}.

We can now add arrows from the nodes m'_{t_0} into a cluster of C of C nodes from which emerges an arrow whose term is $\mathrm{term}(n)$. Observe that we have not omitted nodes, but have simply added penetrator nodes on S and C strands.

Since n is negative, there is a unique incoming arrow $\rightarrow n$. By graph operations we can replace $\rightarrow n$ with the arrow emerging from the cluster C_t. The resulting graph has no cycles, and no lonely or gregarious positive nodes are created by this graph operation. In the new bundle, the nodes m and n are not connected by any path which has an intermediate regular node. These operations add a new set of nodes A to the graph, but each of these new nodes can only be reached from below by paths which traverse m.

To show that any efficient bundle has an equivalent efficient normal bundle, it suffices to show that the graph operations used to eliminate redundancies in Proposition 3.2 preserve efficiency. The only graph operation which might destroy efficiency is adding a message transmission edge between two nodes. However, these nodes are connected in the original bundle by a path which only traverses penetrator nodes. Thus no new paths connecting a regular node to a shadowed node can appear in the modified graph. ■

Proposition 3.23 *Suppose $\mathcal{C}$ is a normal efficient bundle and $(p, \mathcal{L})$ and $(p', \mathcal{L}')$ are transformation paths in $\mathcal{C}$. Assume p is a penetrator path which starts at a simple term, p' is a penetrator path which ends at a simple term, and there is some regular node m such that $\ell(p) \prec m \prec p'_1$. Then for all i with $1 \leq i \leq |p|$ and j with $1 \leq i \leq |p'|$, $\mathcal{L}_i \neq \mathcal{L}'_j$.*

PROOF. By considering the transformation path $(p, \mathcal{L})$ restricted to the integer interval $[1 \ldots i]$ and the transformation path $(p', \mathcal{L}')$ restricted to the integer interval $[j \ldots |p'|]$ we may assume without loss of generality that $i = |p|$ and $j = 1$.

By Proposition 3.20, there are indices α, β such that $\mathrm{term}(p_\alpha) = \mathcal{L}_{|p|}$ and $\mathrm{term}(p'_\beta) = \mathcal{L}'_1$. In particular, $p_\alpha \prec m \prec p'_\beta$ and $\mathrm{term}(p_\alpha)$, $\mathrm{term}(p'_\beta)$ both have single components. Therefore, by bundle efficiency, $\mathrm{term}(p_\alpha) \neq \mathrm{term}(p'_\beta)$. In particular, $\mathcal{L}'_1 \neq \mathcal{L}_{|p|}$. ■

4 A Method for Authentication

In this section we describe our method for establishing authentication results. We first show how to establish whether keys are accessible to the penetrator or not (Section 4.1). We then introduce the notion of a *transformed edge*, in which

a value is sent out and later received in a new component, and the notion of a *transforming edge*, in which a value is received and later sent out in a new component. We define three kinds of authentication tests, and state a theorem about each one, showing what other regular nodes must exist in a bundle, if that bundle contains an example of an authentication test. We will illustrate the first authentication test result using the Needham-Schroeder and Needham-Schroeder-Lowe protocols. Proofs are gathered in Section 4.3, after the main ideas have been explained and illustrated.

In the next section (Section 5), we will apply these authentication test theorems to additional examples. A surprising amount of protocol verification and discovery of counterexamples can be derived directly from the results of the current section.

4.1 Penetrable Keys and Safe Keys

Given a strand space Σ, we can inductively define the set of keys that may become known to the penetrator. We use the relation $\sqsubset_{\mathfrak{K}}$ defined in Definition A.8; $t_0 \sqsubset_{\mathfrak{K}} t$ means that t_0 occurs as a subterm of t in a position where all encryptions surrounding it use keys $K \in \mathfrak{K}$. Thus, either t can be constructed from t_0 simply by (possibly repeated) concatenation, or else t can be written in the form

$$\cdots \{\!|\ \cdots\ t_0\ \cdots\ |\!\}_K\ \cdots$$

where $K \in \mathfrak{K}$ and the dots hide only concatenations and other encryptions with keys in $\mathfrak{K}$. The set $\mathfrak{K}^{-1}$ means the set of inverses of keys in $\mathfrak{K}$. For instance, let $S = \{K_B\} = \{K_B^{-1}\}^{-1}$. Then $N_a \sqsubset_S N_b \{\!|N_a\ A|\!\}_{K_B}$. Moreover, $N_b \sqsubset_\emptyset N_b \{\!|N_a\ A|\!\}_{K_B}$.

In the base case of this definition we refer to $\mathsf{K}_\mathcal{P}$, which is the set of keys known to the penetrator initially, apart from any protocol activity (Definition A.9).

Definition 4.1 *Let* $\mathsf{P}_0 = \mathsf{K}_\mathcal{P}$.

Let $\mathsf{P}_{i+1} = \mathsf{P}_i \cup Y$, *where* $K \in Y$ *if and only if there exists a positive regular node* $n \in \Sigma$ *and a term* t *such that:*

1. t *is a new component of* n, *and*
2. $K \sqsubset_{\mathsf{P}_i{}^{-1}} t$

$\mathsf{P} = \bigcup_i \mathsf{P}_i$.

Thus, either a penetrable key is already penetrated ($\mathsf{K}_\mathcal{P}$), or else some regular strand puts it in a form that could allow it to be penetrated, because for each key protecting it, the matching decryption key is already penetrable. The justification for this definition is that any key that becomes available to the penetrator in any bundle is in fact a member of P.

Proposition 4.2 *Let* $\mathcal{C}$ *be a bundle with* $n \in \mathcal{C}$ *and* $term(n) = K$. *Then* $K \in \mathsf{P}$.

The proof is contained in Section 4.3.1. P is a conservative approximation. It may be larger than the set of keys that the penetrator can really capture, because the strand that would put the key in danger may not be contained in any bundle.

Definition 4.3 *Let S_0 be the set of keys K such that $K \notin \mathsf{K}_\mathcal{P}$ and there is no positive regular node $n \in \Sigma$ and term t such that t is a new component of n and $K \sqsubset t$.*

Let S_{i+1} be the set of keys K such that $K \notin \mathsf{K}_\mathcal{P}$, and for every positive regular node $n \in \Sigma$ and new component t of n, every occurrence of K in t lies within an encryption using some key K_0 where $K_0^{-1} \in \mathsf{S}_i$:

$$\cdots \{\!|\ \cdots\ K\ \cdots\ |\!\}_{K_0}\ \cdots$$

$\mathsf{S} = \bigcup_i \mathsf{S}_i$. *When $K \in \mathsf{S}$, we say that K is* safe *in Σ.*

Evidently, the set of safe keys is disjoint from P. However, there are strand spaces Σ in which there are keys K such that $K \notin \mathsf{P} \cup \mathsf{S}$.

In practice, protocol secrecy goals frequently amount to showing that certain keys are in either S_0 or S_1. Larger values of i seem rarely to occur in these protocols. Showing that a private key or a long-term symmetric key is in S_0 typically reduces to checking that it is assumed not to be in $\mathsf{K}_\mathcal{P}$, because protocols generally avoid emitting terms containing these keys.

For instance, in the Needham-Schroeder protocol, if n is a regular node, then $K \not\sqsubset \mathrm{term}(n)$. Hence, $\mathsf{S}_0 = \mathsf{K} \setminus \mathsf{K}_\mathcal{P}$, which says that any key not initially known to the penetrator is permanently safe.

Many protocols expect session keys to be generated by a key server, which sends them encrypted in the long-term keys of two principals, and no principal ever re-encrypts a session key under a new key. In a particular session, a session key K may be sent encrypted with long term keys are not in $\mathsf{K}_\mathcal{P}$ (or, if they are asymmetric, their inverses are not in $\mathsf{K}_\mathcal{P}$). If the server never re-sends the same session key K in a different session, we can infer that $K \in \mathsf{S}_1$. This idea is illustrated in Sections 5.1 and 5.2.

There also exist protocols in which the session key is translated, in the sense that it is sent out originally encrypted with one key and is later re-encrypted by another principal under a new key. These protocols can also be correct, although they demand special care. The TMN protocol is a (flawed) example [22]. In the case of a correct protocol of this form, it may be necessary to show that the session key is in S_2. However, the fact that S_0 and S_1 cover typical protocols makes this method for proving secrecy particularly easy to use.

It is also easy to prove that a non-key data value such as a nonce is kept secret in some run of a protocol; one simply shows that every term containing it is of the form $\{|h|\}_K$ where $K^{-1} \in \mathsf{S}_i$. Again, typically $i = 0$ or 1. The case $i = 0$ is sufficient to show that the Needham-Schroeder nonces N_a and N_b remain secret, assuming that they are uniquely originating, and assuming that the private keys $K_A^{-1}, K_B^{-1} \notin \mathsf{K}_\mathcal{P}$.

4.2 The Authentication Tests

Fix some strand space Σ. We identify segments of regular strands called *tests* whose presence will guarantee the existence of other regular strands in the bundle; they are strands with transforming edges operating on the test component.

Definition 4.4 *$t = \{|h|\}_K$ is a* test component *for a in n if:*

1. *$a \sqsubset t$ and t is a component of n;*
2. *The term t is not a proper subterm of a component of any regular node $n' \in \Sigma$.*

The edge $n_0 \Rightarrow^+ n_1$ is a test *for a if a uniquely originates at n_0 and $n_0 \Rightarrow^+ n_1$ is a transformed edge for a.*

Clause 2 in the definition of test component ensures that the penetrator cannot get any benefit from building a larger term to send to a regular participant, who might then emit some new message of value to the penetrator.

For instance, in the Needham-Schroeder protocol, if $s_r \in \mathrm{NSResp}[A, B, N_a, N_b]$, then $\{|N_a\, N_b|\}_{K_A}$ is a test component for N_b in $\langle s_r, 2\rangle$, because $\mathrm{term}(\langle s_r, 2\rangle) = \{|N_a\, N_b|\}_{K_A}$ and this component does not occur as a proper subterm of any other regular node. Assuming that the responder B chooses N_b to be uniquely originating at $\langle s_r, 2\rangle$, the edge $\langle s_r, 2\rangle \Rightarrow \langle s_r, 3\rangle$ is a test for N_b.

Tests can use their test components in at least two different ways. If the uniquely originating value is sent in encrypted form, and the challenge is to decrypt it, then that is an outgoing test. If it is received back in encrypted form, and the challenge is to produce that encrypted form, then that is an incoming test. These two kinds of test are illustrated in Figure 15.

Definition 4.5 *The edge $n_0 \Rightarrow^+ n_1$ is an* outgoing test *for a in $t = \{|h|\}_K$ if it is a test for a in which: $K^{-1} \notin \mathsf{P}$; a does not occur in any component of n_0 other than t; and t is a test component for a in n_0.*

The edge $n_0 \Rightarrow^+ n_1$ is an incoming test *for a in $t_1 = \{|h|\}_K$ if it is a test for a in which $K \notin \mathsf{P}$ and t_1 is a test component for a in n_1.*

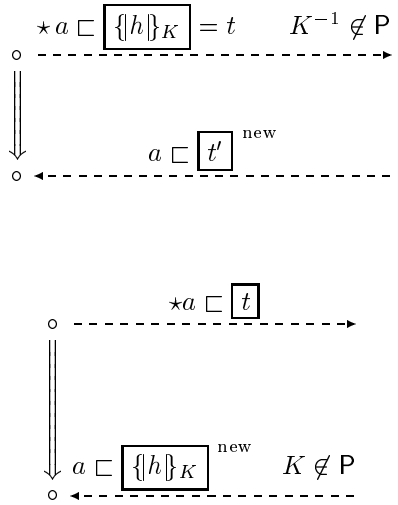
If $K_A^{-1} \notin \mathsf{K}_\mathcal{P}$ (hence $K_A^{-1} \in \mathsf{S}_0$), then the edge $\langle s_r, 2\rangle \Rightarrow \langle s_r, 3\rangle$ is an outgoing test for N_b in $\{|N_a\, N_b|\}_{K_A}$. It is not an incoming test for N_b in $\{|N_b|\}_{K_B}$, because the public key K_B is presumably in $\mathsf{K}_\mathcal{P}$.

The three authentication test results that follow give a powerful method for establishing the authentication goals of protocols. The results with their proofs appear in Section 4.3.2 as Propositions 4.9–4.11.

4.2.1 The Outgoing Authentication Test

Authentication Test 1 *Let $\mathcal{C}$ be a bundle with $n' \in \mathcal{C}$, and let $n \Rightarrow^+ n'$ be an outgoing test for a in t.*

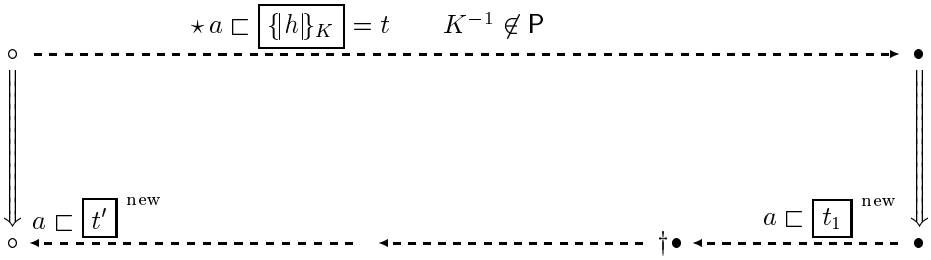
1. *There exist regular nodes $m, m' \in \mathcal{C}$ such that t is a component of m and $m \Rightarrow^+ m'$ is a transforming edge for a.*



$\star$ means a originates uniquely here

$\boxed{t}$ means t is a component of this node

Figure 15: Outgoing and Incoming Tests



• means this regular node must exist

† with assumptions on t_1

Figure 16: Authentication Provided by an Outgoing Test

2. *Suppose in addition that a occurs only in component $t_1 = \{|h_1|\}_{K_1}$ of m', that t_1 is not a proper subterm of any regular component, and that $K_1^{-1} \notin \mathsf{P}$. Then there is a negative regular node m'' with t_1 as a component.*

The meaning of this assertion is illustrated in Figure 16. In this diagram, the two nodes marked $\circ$ represent n and n'. The result assumes that a originates uniquely here (shown by the $\star$), and that the decryption key K^{-1} is safe. The diagram does not represent the assumption that t not be a proper subterm of any regular component, which being non-local is hard to display. The test establishes that $\mathcal{C}$ also contains regular nodes m and m' (marked $\bullet$ at right) with a transforming edge for a. With the assumptions on t_1 given in clause 2, there is also a negative regular node m'', shown with a $\bullet$ on the bottom line, of which t_1 is a component.

Outgoing Tests: The Needham-Schroeder Illustration We may illustrate the outgoing authentication tests by Needham-Schroeder. Assume that $\mathcal{C}$ is a bundle, and the $\mathcal{C}$-height of $s_r \in \text{NSResp}[A, B, N_a, N_b]$ is 3, which means that all three nodes of s_r belong to $\mathcal{C}$. Assume that $K_A^{-1} \notin \mathsf{K}_{\mathcal{P}}$. Finally, assume that N_b originates uniquely, and $N_b \neq N_a$ (which together mean that N_b originates uniquely at $\langle s_r, 2\rangle$).

It follows that the edge $\langle s_r, 2\rangle \Rightarrow \langle s_r, 3\rangle$ is an outgoing test for N_b in $\{|N_a\, N_b|\}_{K_A}$. By Authentication Test 1, there exist regular nodes $m, m' \in \mathcal{C}$ such that $\{|N_a\, N_b|\}_{K_A}$ is a component of m and $m \Rightarrow^+ m'$ is a transforming edge for a. The only negative regular node containing a component of this form is $\langle s_i, 2\rangle$ for $s_i \in \text{NSInit}[A, B', N_a, N_b]$ and some responder B'. Thus, the transforming edge $m \Rightarrow^+ m'$ must be $\langle s_i, 2\rangle \Rightarrow^+ \langle s_i, 3\rangle$, and s_i has $\mathcal{C}$-height 3.

Unfortunately, we have not proved that $s_i \in \text{NSInit}[A, B, N_a, N_b]$ for the expected responder B, rather than some other responder B'. And Figure 2 is a counterexample in which $B' = P \neq B$. Hence we have uncovered a limitation in the authentication achieved by Needham-Schroeder, first noted by Lowe [11, 12],

which led Lowe to amend the protocol to contain the responder's name B in the second message $\{|N_a\, N_b\, B|\}_{K_A}$.

Needham-Schroeder-Lowe Let us next consider a strand space Σ in which the regular strands are:

- For $s_i \in \mathrm{NSLInit}[A, B, N_a, N_b]$, traces of the form:

$$\langle +\{|N_a\, A|\}_{K_B},\quad -\{|N_a\, N_b\, B|\}_{K_A},\quad +\{|N_b|\}_{K_B}\rangle$$

- For $s_r \in \mathrm{NSLResp}[A, B, N_a, N_b]$, traces of the form:

$$\langle -\{|N_a\, A|\}_{K_B},\quad +\{|N_a\, N_b\, B|\}_{K_A},\quad -\{|N_b|\}_{K_B}\rangle$$

To be precise, let $\mathsf{T}_{\mathrm{name}}$ be a distinguished set within A with $\mathsf{T}_{\mathrm{name}} \subset \mathsf{T}$. $\mathrm{NSLInit}[A, B, N_a, N_b]$ and $\mathrm{NSLResp}[A, B, N_a, N_b]$ are empty unless $A, B \in \mathsf{T}_{\mathrm{name}}$, $N_a, N_b \in \mathsf{T}$ but $N_a, N_b \notin \mathsf{T}_{\mathrm{name}}$. In addition, we assume that $\mathrm{NSLResp}[A, B, N_a, N_b]$ is empty unless $N_b \neq N_a$. The correctness of the protocol depends on the assumption that the "public key of" mapping $f : A \mapsto K_A$ is injective.

Assume that $\mathcal{C}$ is a bundle, and the $\mathcal{C}$-height of $s_r \in \mathrm{NSLResp}[A, B, N_a, N_b]$ is 3. Assume that $K_A^{-1} \notin \mathsf{K}_{\mathcal{P}}$. Finally, assume that N_b originates uniquely, and $N_b \neq N_a$ (which together mean that N_b originates uniquely at $\langle s_r, 2\rangle$).

As before, it follows that the edge $\langle s_r, 2\rangle \Rightarrow \langle s_r, 3\rangle$ is an outgoing test for N_b in $\{|N_a\, N_b\, B|\}_{K_A}$. By Authentication Test 1, there exist regular nodes $m, m' \in \mathcal{C}$ such that $\{|N_a\, N_b\, B|\}_{K_A}$ is a component of m and $m \Rightarrow^+ m'$ is a transforming edge for a. The only negative regular node containing a component of this form is $\langle s_i, 2\rangle$ for $s_i \in \mathrm{NSLInit}[A, B, N_a, N_b]$.

Thus, the transforming edge $m \Rightarrow^+ m'$ must be $\langle s_i, 2\rangle \Rightarrow^+ \langle s_i, 3\rangle$, and s_i has $\mathcal{C}$-height 3. This proves that the responder successfully authenticates the initiator in Needham-Schroeder-Lowe.

We will also prove the initiator's authentication guarantee. The proof is very similar, except that it is necessary to use the second part of Authentication Test 1 as well as the first part of it. We include it to illustrate the use of this proof method.

Let $\mathcal{C}$ be a bundle in Σ, and s_i be an initiator's strand in $\mathrm{NSLInit}[A, B, N_a, N_b]$ with $\mathcal{C}$-height 3. Assume $K_A^{-1}, K_B^{-1} \notin \mathsf{K}_{\mathcal{P}}$, and suppose that N_a is uniquely originating.

The edge $\langle s_i, 1\rangle \Rightarrow \langle s_i, 2\rangle$ is an outgoing test for N_a in $\{|N_a\, A|\}_{K_B}$, so it follows (by Authentication Test 1) that there is a regular transforming edge $m \Rightarrow^+ m'$ in $\mathcal{C}$ with $\{|N_a\, A|\}_{K_B}$ a component of the negative node m. This implies that m, m' are the first two nodes of a responder strand $s_r \in \mathrm{NSLInit}[A, B, N_a, N]$. In this step, we used the assumption that $K_B^{-1} \notin \mathsf{K}_{\mathcal{P}}$, from which it follows that $K_B^{-1} \notin \mathsf{P}$.

However, we cannot yet be sure whether $N = N_b$. To infer that B has sent out the same nonce N_b that A eventually receives, we use Part 2 of Authentication Test 1. It implies that $\{|N_a\, N\, B|\}_{K_A}$ is a component of some

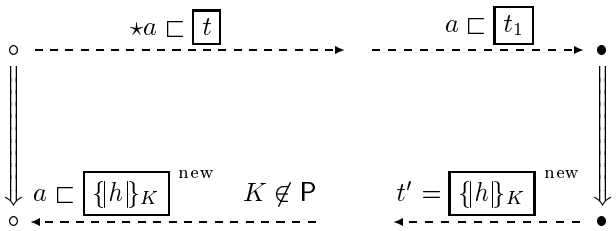


Figure 17: Authentication Provided by an Incoming Test

negative regular node m''. However, m'' can only be $\langle s'_i, 2\rangle$ for some $s'_i \in \text{NSLInit}[A, B, N_a, N]$, since only the second node of an initiator strand receives a component of this form. By the form of an initiator strand, N_a originates at $\langle s'_i, 1\rangle$. Since N_a is uniquely originating, it follows that $\langle s'_i, 1\rangle = \langle s_i, 1\rangle$, so $s'_i = s_i$ and $N = N_b$. In this step, we used the assumption that $K_A^{-1} \notin \mathsf{K}_\mathcal{P}$, from which it follows that $K_A^{-1} \notin \mathsf{P}$.

Thus, we have shown that $\mathcal{C}$ contains a responder strand

$$s_r \in \text{NSLResp}[A, B, N_a, N_b]$$

with $\mathcal{C}$-height 2. This proves that the initiator successfully authenticates the responder in Needham-Schroeder-Lowe.

4.2.2 The Incoming Authentication Test

An authentication test result for incoming tests can be used to infer the existence of a regular transforming edge in protocols in which a nonce is emitted in plaintext, for instance as a challenge, and later received in encrypted form.

Authentication Test 2 *Let $\mathcal{C}$ be a bundle with $n' \in \mathcal{C}$, and let $n \Rightarrow^+ n'$ be an incoming test for a in t'. Then there exist regular nodes $m, m' \in \mathcal{C}$ such that t' is a component of m' and $m \Rightarrow^+ m'$ is a transforming edge for a.*

The meaning of this assertion is illustrated in Figure 17 using the same conventions as in Figure 16. We will apply the incoming authentication test in Sections 5.2 and 5.3.

Although in this paper we will make no use of it, the outgoing and incoming authentication tests also establish an ordering on the nodes, as n occurs before m and m', while n' occurs after. The nodes are ordered $n \prec m \prec m' \prec n'$ in the causal ordering given in Definition A.5. The principal executing n and n' can regard a session key generated at m' as "fresh," because it was created more recently than the beginning of his current run.

The authentication tests are also valid when n and n' are not actually on the same strand, but n is a node known to be in a bundle and to have uniquely originated the test value a, and n' is a node on a different strand that later receives a in transformed form.

4.2.3 The Unsolicited Authentication Test

The authentication property achieved by an unsolicited test is less informative, but frequently useful, for instance when a key server authenticates its clients. We will illustrate authentication via unsolicited tests in Sections 5.1–5.3.

Definition 4.6 *A negative node n is an* unsolicited test *for $t = \{|h|\}_K$ if t is a test component for any a in n and $K \notin \mathsf{P}$.*

Authentication Test 3 *Let $\mathcal{C}$ be a bundle with $n \in \mathcal{C}$, and let n be an unsolicited test for $t = \{|h|\}_K$. Then there exists a positive regular node $m \in \mathcal{C}$ such that t is a component of m.*

4.3 Proving the Method for Authentication Correct

In this section we will justify our method for establishing authentication results. We first prove Proposition 4.2, justifying our treatment of secrecy. We then prove theorems establishing the three kinds of authentication test which so many protocols use. Each authentication test establishes the existence of regular nodes, typically forming a transforming edge (Section 4.3.2).

4.3.1 Keys Available to the Penetrator are Penetrable

Proposition 4.7 *Let $\mathcal{C}$ be a bundle with $n \in \mathcal{C}$ and $\mathrm{term}(n) = K$. Then $K \in \mathsf{P}$.*

Proof. By Propositions 3.8, 3.14, and 3.22, we may assume that $\mathcal{C}$ is normal, that it has simple bridges, and that it is efficient. We may assume that n is positive. We argue by induction on the well-founded relation $\prec_{\mathcal{C}}$. Our induction hypothesis is that, for all $n' \prec_{\mathcal{C}} n$, $\mathrm{term}(n') \in \mathsf{K}$ implies $\mathrm{term}(n') \in \mathsf{P}$.

By Proposition 3.18, we may let $(p, \mathcal{L})$ be a transformation path such that $\ell(p) = n$, K originates at p_1, and $K \sqsubset \mathcal{L}_i$ for all i with $1 \leq i \leq |p|$. If $|p| = 1$ and n is a penetrator node, then n is a K node, so $K \in \mathsf{K}_{\mathcal{P}}$. Otherwise, because $\mathcal{C}$ is normal and efficient, p_1 is not a penetrator node (which could only be a K node).

Let p_λ be the last regular node on p. The penetrator path

$$p_\lambda \longmapsto \cdots \longmapsto \ell(p)$$

is a falling path traversing no D strand key edges. By the induction hypothesis, if p traverses a D strand s, then the key edge entering contains a key $K_0 \in \mathsf{P}$. By Proposition 3.11, $K \sqsubset_{\mathsf{P}^{-1}} \mathcal{L}_\lambda$, where $\mathcal{L}_\lambda$ is the distinguished component of p_λ. By Proposition 3.20, there is a j such that $\lambda \leq j \leq |p|$ such that $p_j = \mathcal{L}_j = \mathcal{L}_\lambda$. Since p_λ is a positive regular node, it remains to be shown that $\mathcal{L}_\lambda$ occurs *new* on some positive regular node.

Let κ be the least index such that $\mathcal{L}_i = \mathcal{L}_\lambda$ for all i for $\kappa \leq i \leq \lambda$. By definition, p_κ is positive. By the efficiency of $\mathcal{C}$ and Proposition 3.23, there is no i with $\kappa \leq i < \lambda$ such that p_i is simple. By Proposition 3.20, p_κ is not a penetrator node, which would be a D or E strand and thus have a simple positive node. Therefore p_κ is a regular node. $\mathcal{L}_\lambda$ is a new component of p_κ by the choice of κ. ■

4.3.2 Proofs of the Authentication Tests

Proposition 4.8 *Suppose $(p, \mathcal{L})$ is a key edge free transformation path such that p_1 and $\ell(p)$ are regular and $\mathcal{L}_1 \neq \mathcal{L}_{|p|}$.*

1. *Let $\mathcal{L}_1 = \{|h_1|\}_{K_1}$. Suppose that $\mathcal{L}_1$ is not a proper subterm of any regular component, and suppose that $K_1^{-1} \notin \mathsf{P}$. Then the smallest index α such that $\mathcal{L}_\alpha \neq \mathcal{L}_{\alpha+1}$ is such that p_α is regular. Moreover, $p_\alpha \Rightarrow^+ p_{\alpha+1}$ is a transforming edge.*

2. *Let $\mathcal{L}_{|p|} = \{|h_\lambda|\}_{K_\lambda}$. Suppose that $\mathcal{L}_{|p|}$ is not a proper subterm of any regular component, and suppose that $K_\lambda \notin \mathsf{P}$. Then the largest index α such that $\mathcal{L}_\alpha \neq \mathcal{L}_{\alpha-1}$ is such that p_α is regular. Moreover, $p_{\alpha-1} \Rightarrow^+ p_\alpha$ is a transforming edge.*

PROOF. We prove item 1. The proof of 2 is analogous. Suppose p_α is not regular. Then $p_\alpha \Rightarrow^+ p_{\alpha+1}$ lies either on a D-strand or an E-strand.

In the D-strand case $\mathrm{term}(p_\alpha) = \{|h_1|\}_{K_1}$. But $p_\alpha \Rightarrow^+ p_{\alpha+1}$ cannot lie on a D-strand because we have assumed that $K_1^{-1} \notin \mathsf{P}$.

So suppose that $p_\alpha \Rightarrow^+ p_{\alpha+1}$ lies on an E-strand, in which case $\mathcal{L}_\alpha$ is a proper subterm of $\mathcal{L}_{\alpha+1} = \mathrm{term}(p_{\alpha+1})$. Since $\mathcal{C}$ is normal and $p_\alpha \Rightarrow^+ p_{\alpha+1}$ is constructive, every penetrator edge between p_α and the next regular node p_β on p, which exists since $\ell(p)$ is regular, is constructive.

By Proposition 3.10, the path $p_\alpha \longmapsto \cdots \longmapsto p_\beta$ is rising, so $\mathcal{L}_1 = \mathcal{L}_\alpha$ is a proper subterm of $\mathcal{L}_{\alpha+1}$ which in turn is a subterm of $\mathrm{term}(p_\beta)$. This contradicts the assumption that $\mathcal{L}_1$ is not a proper subterm of any regular component.

$p_\alpha \Rightarrow^+ p_{\alpha+1}$ is a transforming edge because $\mathcal{L}_{\alpha+1}$ is a new component on the strand of $p_{\alpha+1}$. ■

Proposition 4.9 *Let $\mathcal{C}$ be a normal bundle with $n' \in \mathcal{C}$, and let $n \Rightarrow^+ n'$ be an outgoing test for a in t. Then there exist regular nodes $m, m' \in \mathcal{C}$ such that t is a component of m and $m \Rightarrow^+ m'$ is a transforming edge for a.*

Suppose in addition that a occurs only in component $t_1 = \{|h_1|\}_{K_1}$ of m'. Suppose that t_1 is not a proper subterm of any regular component, and suppose that $K_1^{-1} \notin \mathsf{P}$. Then there is a negative regular node with t_1 as a component.

PROOF. Because $n \Rightarrow^+ n'$ is a transformed edge for a, there is a new component t' of n' with $a \sqsubset t'$.

By Proposition 3.18, there is a transformation path $(p, \mathcal{L})$ in $\mathcal{C}$ with $p_1 = n$, $\ell(p) = n'$, $\mathcal{L}_{|p|} = t'$, and $a \sqsubset \mathcal{L}_i$ for all i. Since t' is new in n', $\mathcal{L}_1 \neq t'$. In fact, because a occurs in no component of n other than t, $\mathcal{L}_1 = t$. In particular, $\mathcal{L}_1 \neq \mathcal{L}_{|p|}$.

By the first part of Proposition 4.8, the smallest index α such that $\mathcal{L}_\alpha \neq \mathcal{L}_{\alpha+1}$ is such that p_α is regular. Moreover, $p_\alpha \Rightarrow^+ p_{\alpha+1}$ is a transforming edge. It follows that $t = \mathcal{L}_1 = \mathcal{L}_\alpha$ is a component of $m = p_\alpha$.

Consider now the additional assumptions on the components of $m' = p_{\alpha+1}$. Since $\mathcal{L}_{\alpha+1}$ is a component of $\mathrm{term}(m')$ that contains a as subterm and a occurs only in component $t_1 = \{|h_1|\}_{K_1}$, $\mathcal{L}_{\alpha+1} = t_1$.

If $t_1 = t'$, then n' itself is a negative regular node with t_1 as a component. Otherwise, apply Proposition 4.8 again to conclude that smallest index $\beta > \alpha+1$ such that $\mathcal{L}_\beta \neq \mathcal{L}_{\beta+1}$ is such that p_β regular. Now $t_1 = \mathcal{L}_{\alpha+1} = \mathcal{L}_\beta$ is a component of p_β. ■

Proposition 4.10 *Let $\mathcal{C}$ be a normal bundle with $n' \in \mathcal{C}$, and let $n \Rightarrow^+ n'$ be an incoming test for a in t'. Then there exist regular nodes $m, m' \in \mathcal{C}$ such that t' is a component of m' and $m \Rightarrow^+ m'$ is a transforming edge for a.*

PROOF. By Proposition 3.18, there is a transformation path $(p, \mathcal{L})$ in $\mathcal{C}$ with $p_1 = n$, $\ell(p) = n'$, $\mathcal{L}_{|p|} = t'$, and $a \sqsubset \mathcal{L}_i$ for all i. Since t' is new in n', $\mathcal{L}_1 \neq t'$. In particular, $\mathcal{L}_1 \neq \mathcal{L}_{|p|}$.

By the second part of Proposition 4.8 , the largest index α such that $\mathcal{L}_\alpha \neq \mathcal{L}_{\alpha-1}$ is such that $p_{\alpha-1}$ is regular. Moreover, $p_{\alpha-1} \Rightarrow^+ p_\alpha$ is a transforming edge. In particular $t' = \mathcal{L}_{|p|} = \mathcal{L}_\alpha$ is a component of $m' = p_\alpha$.■

Proposition 4.11 *Let $\mathcal{C}$ be a normal bundle with $n \in \mathcal{C}$, and let n be an unsolicited test for $t = \{|h|\}_K$. Then there exists a positive regular node $m \in \mathcal{C}$ such that t is a component of m.*

PROOF. By Proposition 3.18, there is a key edge free transformation path $(p, \mathcal{L})$ in $\mathcal{C}$ with $p_1 = n$, $\ell(p) = n'$, $\mathcal{L}_{|p|} = t$, $t \sqsubset \mathcal{L}_i$ for all i and such that t originates at p_1.

Since t originates at p_1, p_1 is a positive node. We claim p_1 is a regular node. Suppose otherwise. Since $t \sqsubset p_k$, p_k is neither an M-node nor a K-node. Since t originates at p_1, p_1 cannot be a S-node, a C-node nor a D-node.

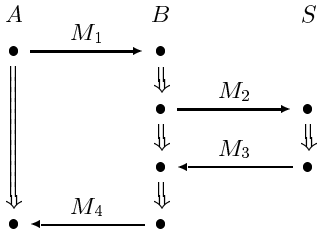
If p_1 is a E-node, then p_1 is the positive ciphertext (last) node on a E-strand. Since $K \notin \mathsf{P}$, t is a proper subterm of term(p_1). Hence t is a subterm of the plaintext (first) node on the strand, so t cannot originate at p_1 in this case either.

Therefore, p_1 must be a regular node as claimed. By the definition of test component, t is not a proper subterm of any component of p_1, so t is a component of p_1. ■

5 Protocol Correctness and Protocol Failure

In this section we apply the authentication theorems of Section 4.2 to several additional examples. They are the Otway-Rees protocol [18, 1, 24], the Neuman-Stubblebine protocol [17, 23], and the Wool-Lam protocol [25, 26]. We do so to illustrate the ease and directness with which these theorems lead to authentication results.

It is remarkably easy to find the outgoing, incoming, and unsolicited tests that provide a protocol's authentication guarantees, assuming that the protocol does not allow its test components to occur in nested contexts. That would violate Clause 2 of the definition of test component (Definition 4.4). The method works for public-key protocols, and for shared symmetric key protocols also.



$M_1 = M\ A\ B\ \{|N_a\ M\ A\ B|\}_{K_{AS}}$

$M_2 = M\ A\ B\ \{|N_a\ M\ A\ B|\}_{K_{AS}}\ \{|N_b\ M\ A\ B|\}_{K_{BS}}$

$M_3 = M\ \{|N_a\ K_{AB}|\}_{K_{AS}}\ \{|N_b\ K_{AB}|\}_{K_{BS}}$

$M_4 = M\ \{|N_a\ K_{AB}|\}_{K_{AS}}$

Figure 18: Message Exchange in Otway-Rees

In the Otway-Rees protocol, each of the initiator and the responder uses an outgoing test to authenticate a server strand. The server uses an unsolicited test to establish that the initiator and responder have each sent a message.

The Neuman-Stubblebine protocol uses a combination of incoming tests and unsolicited tests. It is a two-part protocol: in the first part the initiator and responder use a key distribution server to authenticate one another and acquire a session key. In the second part the key distribution server is not involved; the initiator re-presents a ticket obtained in a run of part I, and the initiator and responder re-authenticate one another. The first part is valid in itself [23] (ignoring an implausible type-flaw attack [10]). The second part is flawed, both in itself [10] and in undermining the guarantees that part I provide in isolation [23]. We will use the authentication test results to explain both why the first part works in isolation, and also why the addition of the second part undermines its guarantees.

5.1 The Otway-Rees Protocol

The Otway-Rees protocol (Figure 18) uses long-term symmetric keys shared with a key server to distribute a new session key for a conversation between two clients. The protocol does not establish that the same key is delivered to both A and B [24], only that if either A or B reaches the end of its strand, then the other has submitted the expected matching original request $\{|N_b\ M\ A\ B|\}_{K_{BS}}$ or $\{|N_a\ M\ A\ B|\}_{K_{AS}}$. Also, K is not disclosed, assuming that the server chooses a uniquely originating session key K.

5.1.1 Strand Spaces for Otway-Rees

The regular strands are defined to be of the form:

1. "Initiator strands" in $\mathrm{Init}[A, B, N, M, K]$, which have trace:

$$\langle + M\ A\ B\ \{|N\ M\ A\ B|\}_{K_{AS}}, - M\ \{|N\ K|\}_{K_{AS}} \rangle$$

2. 'Responder strands" in $\mathrm{Resp}[A, B, N, M, K, H, H']$, which have trace:

$$\begin{aligned} \langle & - M\ A\ B\ H, \\ & + M\ A\ B\ H\ \{|N\ M\ A\ B|\}_{K_{BS}}, \\ & - M\ H'\ \{|N\ K|\}_{K_{BS}}, \\ & + M\ H' \rangle \end{aligned}$$

3. 'Server strands" in $\mathrm{Serv}[A, B, N_a, N_b, M, K]$ with trace:

$$\begin{aligned} \langle & - M\ A\ B\ \{|N_a\ M\ A\ B|\}_{K_{AS}}\ \{|N_b\ M\ A\ B|\}_{K_{BS}}, \\ & + M\ \{|N_a\ K|\}_{K_{AS}}\ \{|N_b\ K|\}_{K_{BS}} \rangle \end{aligned}$$

The principal active in $\mathrm{Init}[A, B, N, M, K]$ is A, while the active principal in $\mathrm{Resp}[A, B, N, M, K, **]$ is B.[1] We define LT to be the set of long-term keys, i.e. the range of the injective function K_{AS} for $A \in \mathsf{T}_{\mathrm{name}}$. All long-terms keys are symmetrical: $K \in \mathsf{LT}$ implies $K = K^{-1}$.

We will use three side assumptions.

1. We assume that the responder's nonce originates on that strand, which implies that $\mathrm{Resp}[A, B, N, M, K, H, H'] = \emptyset$ if $N \sqsubset H$.

2. We assume that the terms H and H', which are simply forwarded by the responder with no interpretation or processing, contain no proper encrypted subterms. That is, $\{|g|\}_K \sqsubset H$ and $\{|g|\}_K \neq H$ implies

$$\mathrm{Resp}[A, B, N, M, K, H, H'] = \emptyset;$$

   and likewise for H'. We point out below (Section 5.1.3) that this assumption does not mask any possible failure of the protocol.

3. We assume that the server generates keys in a reasonable manner, in the sense that $\mathrm{Serv}[**, K] = \emptyset$ unless: $K \notin \mathsf{K}_{\mathcal{P}}$; $K = K^{-1}$; K is uniquely originating; and $K \notin \mathsf{LT}$. It follows from the unique origination assumption that the cardinality $|\mathrm{Serv}[**, K]| \leq 1$ for every K.

Let Σ be a strand space satisfying these conditions.

[1] We sometimes use an asterisk to indicate a union over a particular argument position, and a double asterisk to indicate a union over all remaining argument positions. Thus, for instance, $\mathrm{Serv}[*, *, *, *, *, K]$ is the set of all server strands emitting the session key K; $\mathrm{Resp}[A, B, N, M, K, **]$ is the set of all responder strands with initiator A, responder B, nonce N, round number M, session key K, and any value of the remaining parameters. We will also abbreviate a form like $\mathrm{Serv}[*, *, *, *, *, K]$ to $\mathrm{Serv}[**, K]$.

5.1.2 Otway-Rees Authentication

Structurally, Otway-Rees achieves its authentication guarantees in three steps.

1. The long-term keys LT are not disclosed by the protocol. Thus, if $K \in \mathsf{LT}$ and $K \notin \mathsf{K}_{\mathcal{P}}$, then $K \in \mathsf{S}_0$. Hence, if the server distributes a session key K' to principals with uncompromised keys, then $K' \in \mathsf{S}_1$.

2. The server strand receives an unsolicited test that authenticates the initial positive node of the initiator and responder.

3. The initiator strand contains an outgoing test for N_a in $\{|N_a\, M\, A\, B|\}_{K_{AS}}$; this authenticates the server strand. Likewise, the responder strand contains an outgoing test for N_b in $\{|N_b\, M\, A\, B|\}_{K_{BS}}$, which authenticates the server strand.

The initiator authenticates the responder only in that it authenticates the server strand, which has authenticated the occurrence of the responder's initial positive node. The situation is symmetrical for the responder authenticating the initiator.

Because $K \not\sqsubset \mathrm{term}(n)$ for long-term keys $K \in \mathsf{LT}$ and regular nodes n, Definition 4.3 immediately entails $\mathsf{LT} \subset \mathsf{S}_0 \cup \mathsf{K}_{\mathcal{P}}$. Because the initiator and responder strands emit no new components in which keys occur, a session key can be compromised only if the server sends it out encrypted with a compromised long term key. By the unique origination assumption on session keys, if it is sent out under uncompromised long term keys, then the server will never re-use it with compromised long term keys. Summarizing this, we have:

Proposition 5.1 $\mathsf{LT} \subset \mathsf{S}_0 \cup \mathsf{K}_{\mathcal{P}}$. *If* $K_{AS}, K_{BS} \notin \mathsf{K}_{\mathcal{P}}$ *and* $\mathrm{Serv}[A, B, *, *, *, K] \neq \emptyset$ *then* $K \in \mathsf{S}_1$.

Turning now to the server's authentication guarantee, we use unsolicited tests.

Proposition 5.2 *Suppose that* $\mathcal{C}$ *is a bundle in* Σ*;* $A \neq B$*;* $K_{AS}, K_{BS} \notin \mathsf{K}_{\mathcal{P}}$*; and* $s \in \mathrm{Serv}[A, B, N_a, N_b, M, *]$ *has* $\mathcal{C}$*-height 1.*

Then there exist $s_i \in \mathrm{Init}[A, B, N_a, M, *]$ *and* $s_r \in \mathrm{Resp}[A, B, N_b, M, **]$ *such that* s_i *has* $\mathcal{C}$*-height 1 and* s_r *has* $\mathcal{C}$*-height 2.*

Proof. The terms $\{|N_a\, M\, A\, B|\}_{K_{AS}}$ and $\{|N_b\, M\, A\, B|\}_{K_{BS}}$ are unsolicited tests, and therefore (Authentication Test 3) occur on positive regular nodes in $\mathcal{C}$. When $A \neq B$, the latter occurs positively only on a node $\langle s_r, 2\rangle$ where $s_r \in \mathrm{Resp}[A, B, N_b, M, **]$.

As for $\{|N_a\, M\, A\, B|\}_{K_{AS}}$, it may occur positively either on a strand $s_i \in \mathrm{Init}[A, B, N_a, M, *]$ or as H or H' in a strand $s'_r \in \mathrm{Resp}[**, H, *]$ or $\mathrm{Resp}[**, H']$. Let S be the set of all regular nodes in $\mathcal{C}$ having $\{|N_a\, M\, A\, B|\}_{K_{AS}}$ as a component. Since S is non-empty, it has a $\preceq_{\mathcal{C}}$-minimal member n_0 (Proposition A.6). Since neither H nor H' occurs new on a responder strand, n_0 can only be of the form $\langle s_i, 1\rangle$ for $s_i \in \mathrm{Init}[A, B, N_a, M, *]$. ■

If $A = B$, then $\{|N\ M\ A\ B|\}_{K_{AS}} = \{|N\ M\ A\ B|\}_{K_{BS}}$, so the server can no longer be sure that both an initiator strand and a responder strand are present. This is the explanation for the odd attack, attributed to Michael Goldsmith, in which "the responder thinks he wants to talk to himself, but he really doesn't."

1. $P(B) \longrightarrow B$: $\quad B\ B\ M\ H$;
2. $B \longrightarrow P(S)$: $\quad B\ B\ M\ H\ \{|N_b\ M\ B\ B|\}_{K_{BS}}$
3. $P(B) \longrightarrow S$: $\quad B\ B\ M \{|N_b\ M\ B\ B|\}_{K_{BS}}\ \{|N_b\ M\ B\ B|\}_{K_{BS}}$

which causes a normal server strand, despite the non-existence of any active initiator.

Proposition 5.3 *Suppose that $\mathcal{C}$ is a bundle in Σ; $A \neq B$; $K_{AS} \notin \mathsf{K}_{\mathcal{P}}$; and $s_i \in \mathrm{Init}[A, B, N_a, M, K]$ has $\mathcal{C}$-height 2.*

Then there exists $s \in \mathrm{Serv}[A, B, N_a, *, M, K]$ with $\mathcal{C}$-height 2.

PROOF. $\langle s_i, 1\rangle \Rightarrow^+ \langle s_i, 2\rangle$ is an outgoing test for N_a in $\{|N_a\ M\ A\ B|\}_{K_{AS}}$. Therefore there is a regular transforming edge for N_a (Authentication Test 1). By inspection, this can only lie on a server strand $s \in \mathrm{Serv}[A, B, N_a, *, M, K]$. ■

Proposition 5.4 *Suppose that $\mathcal{C}$ is a bundle in Σ; $A \neq B$; $K_{BS} \notin \mathsf{K}_{\mathcal{P}}$; and $s_r \in \mathrm{Resp}[A, B, N_b, M, K, **]$ has $\mathcal{C}$-height 3.*

Then there exists $s \in \mathrm{Serv}[A, B, *, N_b, M, K]$ with $\mathcal{C}$-height 2.

PROOF. $\langle s_r, 2\rangle \Rightarrow^+ \langle s_r, 3\rangle$ is an outgoing test for N_b in $\{|N_b\ M\ A\ B|\}_{K_{BS}}$. Therefore there is a regular transforming edge for N_b (Authentication Test 1). By inspection, this can only lie on a server strand $s \in \mathrm{Serv}[A, B, *, N_b, M, K]$. ■

These three theorems exhaust the authentication that this protocol actually achieves. Consider, for example, the initiator's guarantee that the responder has been active in a bundle $\mathcal{C}$ containing a strand s_i in $\mathrm{Init}[A, B, N_a, M, K]$. It follows from Proposition 5.3, which establishes that the bundle contains some $s' \in \mathrm{Serv}[A, B, N_a, *, M, K]$, together with Proposition 5.2, which further shows that some $s_r \in \mathrm{Resp}[A, B, *, M, **]$ has $\mathcal{C}$-height 2. Observe that the Otway-Rees protocol cannot possibly guarantee that the responder strand (even if completed) will receive the same session key [24].

5.1.3 The Constraint on Uninterpreted Terms

In Section 5.1.1, we assumed (Clause 2) that the terms H and H' contain no encrypted proper subterms for a responder strand in $\mathrm{Resp}[A, B, N, M, K, H, H']$. However, the responder B cannot enforce this constraint, because in the intended case, these are terms encrypted in A's long-term key, which are unintelligible to B.

In this section we will check that this unenforceable constraint does not hide any attacks. In particular, if the penetrator can succeed without our restrictive assumption, then he can also succeed if it is in force.

To this end, we modify the specification of the Otway-Rees protocol by removing the restriction in Clause 2 that the terms H and H' contain no encrypted proper subterms. Let us call this new protocol "unconstrained Otway-Rees" to distinguish it form the original protocol, which we will refer to (in this section only) as "constrained Otway-Rees". Note that any constrained Otway-Rees bundle is also an unconstrained Otway-Rees bundle. We then show any unconstrained Otway-Rees bundle $\mathcal{C}'$ is nearly equivalent (in a sense defined below) to a constrained Otway-Rees bundle $\mathcal{C}$.

To facilitate the following discussion, we will refer to the locations of the H and H' subterms of $\mathrm{Resp}[A, B, N, M, K, H, H']$ nodes as *insignificant locations* and the terms at those locations as *insignificant terms.*

Definition 5.5 *A* near equivalence *of unconstrained Otway-Rees strand spaces $\mathcal{C}$ on Σ and $\mathcal{C}'$ on Σ' is a bijection $\mathcal{I}$ from the regular nodes of $\mathcal{C}$ to those of $\mathcal{C}'$ satisfying*

1. *$\mathcal{I}$ preserves the strand structure, that is $m \Rightarrow^+ n$ if and only if $\mathcal{I}(m) \Rightarrow^+ \mathcal{I}(n)$.*
2. *For any regular node $n \in \mathcal{C}$, $\mathrm{term}(n)$ and $\mathrm{term}(\mathcal{I}(n))$ are identical except for insignificant locations of $\mathrm{term}(n)$ and $\mathrm{term}(\mathcal{I}(n))$.*
3. *A simple term originates uniquely on regular nodes in Σ iff it originates uniquely on regular nodes in Σ'.*

This definition is clearly weaker than the notion of equivalence (Definition 2.1) in that the underlying strand spaces of the bundles may be different. Moreover, for regular nodes n and $\mathcal{I}(n)$, the corresponding terms $\mathrm{term}(n)$ and $\mathrm{term}(\mathcal{I}(n))$ may be different.

Proposition 5.6 *Any unconstrained Otway-Rees bundle $\mathcal{C}'$ is nearly equivalent to a constrained Otway-Rees bundle $\mathcal{C}$.*

Proof. Let $H_0, H_0' \in \mathsf{T}$ be fixed values, chosen so that neither originates uniquely in Σ'. Let Σ contain the same initiator and server strands as Σ', and the same penetrator strands, together with countably many M-strands emitting the term H_0 and countably many M-strands emitting the term H_0'. Let the responder strands of Σ be synthesized from those of Σ' be replacing the values of the parameters H and H' by H_0 and H_0'; hence we have a bijection correlating the strands of $\mathrm{Resp}[A, B, N_b, M, K, **]$ in Σ' and $\mathrm{Resp}[A, B, N_b, M, K, H_0, H_0']$ in Σ. By the way we selected H_0 and H_0', Σ satisfies Clause 2.

A term t uniquely originates on a regular strand in Σ' iff it uniquely originates on a regular strand in Σ; likewise, the two strand spaces have the same value for $\mathsf{K}_{\mathcal{P}}$. Hence, clauses 1 and 3 are also satisfied, so Σ satisfies all the conditions for an Otway-Rees strand space.

We may now synthesize a bundle $\mathcal{C}$ in Σ from $\mathcal{C}'$. We include the same initiator, server, and penetrator strands (with the same height). For each responder strand in $\mathrm{Resp}[A, B, N_b, M, K, H, H']$ contained in $\mathcal{C}'$, we include the

correlated strand in $\mathrm{Resp}[A, B, N_b, M, K, H_0, H_0']$, with the same height. We cannot connect these strands directly to the expected sender or recipient, because they require H_0 in place of H and H_0' in place of H'. However, we may use M-strands to emit the newly required values, and S- and C-strands to splice them in the required positions. Similarly, we use S- and C-strands to splice them out again and re-insert the values used in $\mathcal{C}'$ between each responder strand and the rest of the bundle. The resulting bundle $\mathcal{C}$ is a counterexample to the same property in Σ, because these properties are independent of the values of H, H' occurring in their responder strands. The other regular strands are unchanged. ■

Hence we may conclude that a strand space Σ' satisfies the same authentication properties, even if Clause 2 fails in Σ'.

This technique may be applied more generally to prove authentication results for protocols which contain unconstrained terms. Suppose Σ is strand space in which the regular strands are given as traces in parametric form

$$\mathrm{P}[\rho, \vec{A}, H] = \langle \mathrm{P}_1[\rho, \vec{A}, H], \ldots, \mathrm{P}_n[\rho, \vec{A}, H] \rangle$$

where $\vec{A}$ and H range over terms and ρ indicates a protocol role such as server or responder. Assume further that

1. For each i, H occurs only as a component of the term $\mathrm{P}_i[\rho, \vec{A}, H]$,
2. H is allowed to assume any value in the message algebra.

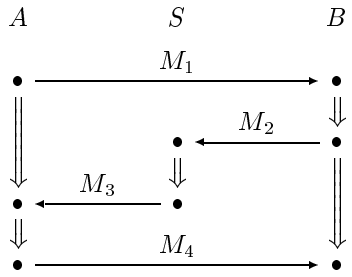
Under these hypotheses, to prove any authentication results we may impose the following constraint on H: $H \in \mathsf{T}$ and H does not occur anywhere else on regular strands.

5.2 Neuman-Stubblebine

The Neuman-Stubblebine protocol [17] contains two sub-protocols. We will call the first sub-protocol the authentication protocol and the second sub-protocol the re-authentication protocol. In the authentication sub-protocol, a key distribution center generates a session key for an initiator (a network client) and a responder (a network server); the message exchange is shown in Figure 19. This session key is embedded in encrypted form in a re-usable ticket of the form $\{\!|A\, K\, T|\!\}_{K_{BS}}$.

Strands of the form shown in the columns labelled A, B, and S in Figure 19 will be called $\mathrm{Init}[A, B, N_a, N_b, t_b, K, H]$, $\mathrm{Resp}[A, B, N_a, N_b, t_b, K]$, and $\mathrm{Serv}[A, B, N_a, N_b, t_b, K]$, respectively.

As in Section 5.1, we define LT to be the set of long-term keys, i.e. the range of the injective function K_{AS} for $A \in \mathsf{T}_{\mathrm{name}}$. All long-terms keys are symmetrical: $K \in \mathsf{LT}$ implies $K = K^{-1}$. We likewise assume that the server generates keys in a reasonable way, meaning that that $\mathrm{Serv}[{*}{*}, K] = \emptyset$ unless: $K \notin \mathsf{K}_{\mathcal{P}}$; $K = K^{-1}$; K is uniquely originating; and $K \notin \mathsf{LT}$. Because of the unique origination assumption, it follows that the cardinality $|\mathrm{Serv}[{*}{*}, K]| \leq 1$ for every K.



$M_1 = A\, N_a$

$M_2 = B\, \{|A\, N_a\, t_b|\}_{K_{BS}}\, N_b$

$M_3 = \{|B\, N_a\, K\, t_b|\}_{K_{AS}}\, \{|A\, K\, t_b|\}_{K_{BS}}\, N_b$

$M_4 = \{|A\, K\, t_b|\}_{K_{BS}}\, \{|N_b|\}_K$

Figure 19: Neuman-Stubblebine Part I (Authentication)

The overall strategy for showing the responder's guarantee, assuming given a strand $s_r \in \mathrm{Resp}[A, B, N_a, N_b, t_b, K]$ with $K_{AS}, K_{BS} \notin \mathsf{K}_{\mathcal{P}}$, is the following:

1. As with Otway-Rees, $\mathsf{LT} \subset \mathsf{S}_0 \cup \mathsf{K}_{\mathcal{P}}$. So for all K', $K' \in \mathsf{S}_1$ whenever $\mathrm{Serv}[A, B, *, *, *, K'] \neq \emptyset$.

2. $\{|A\, K\, t_b|\}_{K_{BS}}$ is an unsolicited test, which can originate only on a regular strand. This can only be a server strand $s_s \in \mathrm{Serv}[A, B, *, *, t_b, K]$. Therefore $K \in \mathsf{S}_1$.

3. $M_2 \Rightarrow M_4$ is an incoming test for N_b in $\{|N_b|\}_K$. Hence there is a regular transforming edge producing $\{|N_b|\}_K$. This can lie only on the second and third nodes of an initiator strand $s_i \in \mathrm{Init}[A', B', N'_a, N_b, t'_b, K, *]$.

4. Since $\langle s_i, 2\rangle$ contains $\{|B'\, N'_a\, K\, t'_b|\}_{K_{A'S}}$ and $K \in \mathsf{S}_1$, it follows that $K^{-1}_{A'S} \notin \mathsf{P}$. Moreover $K^{-1}_{A'S} = K_{A'S}$.

   So $\{|B'\, N'_a\, K\, t'_b|\}_{K_{A'S}}$ is an unsolicited test, which can originate only on a regular strand. This can only be a server strand $s'_s \in \mathrm{Serv}[A', B', N'_a, *, t'_b, K]$.

5. Since server strands construct uniquely originating keys, and K originates on both s_s and s'_s, it follows that $s_s = s'_s$. Hence, $A' = A$, $B' = B$, and $t'_b = t_b$. Therefore, $s_i \in \mathrm{Init}[A, B, *, N_b, t_b, K, *]$, and this strand has height at least three.

The initiator's guarantee is simpler to establish. The edge $M_1 \Rightarrow M_3$ on an initiator strand is an incoming test for N_a in $\{|B\, N_a\, K\, t_b|\}_{K_{AS}}$. It shows there is a server strand $s_s \in \mathrm{Serv}[A, B, N_a, *, t_b, K]$. The first node of s_s is an unsolicited test, showing the existence of a responder strand $s_r \in \mathrm{Resp}[A, B, N_a, *, t_b, *]$.

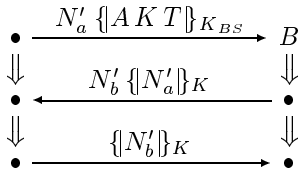


Figure 20: Neuman-Stubblebine, Part II (Re-authentication)

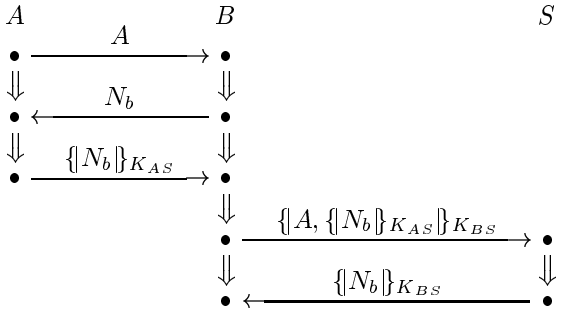


Figure 21: Woo-Lam

In the re-authentication sub-protocol, the key distribution center no longer needs to be involved; the initiator again presents the same ticket to the responder, as shown in Figure 20. However, in the presence of this additional sub-protocol, step 3 in the responder's guarantee can no longer be completed. There is certainly still a transforming edge producing $\{|N_b|\}_K$, but this edge may lie either on an initiator strand for Part I of the protocol, or on (conceivably) either type of strand for Part II. By contrast, the initiator's guarantee for Part I is unaffected, because we have not added any strand with a transforming edge producing a term of the form $\{|B\,N_a\,K\,t_b|\}_{K_{AS}}$.

5.3 The Woo-Lam Protocol

The Woo-Lam one-way authentication protocol [25] also uses an incoming test, although in a flawed way [26, 3, 7]. It is intended to allow an initiator (client) A to authenticate his presence to a responder (networked service) B, by means of long-term keys shared with a key server. A receives no authenticating information about B. The behavior of the protocol is given in Figure 21.

It is clear from Figure 21 how this is *intended* to work. The $\Rightarrow^+$ edge from B's first transmission of N_b to its final reception of $\{|N_b|\}_{K_{BS}}$ is intended to serve as an incoming test with that term as test component. The server's edge $\{|A, \{|N_b|\}_{K_{AS}}|\}_{K_{BS}} \Rightarrow \{|N_b|\}_{K_{BS}}$ is intended as the corresponding transforming edge. It "authenticates" that the server has found N_b inside A's encrypted

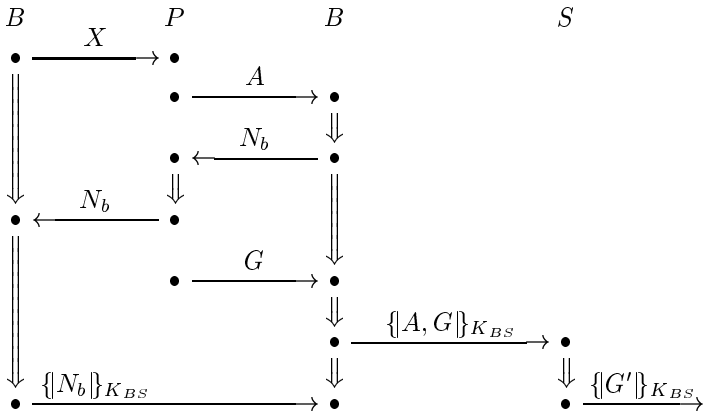


Figure 22: Woo-Lam Infiltrated

message.

Unfortunately this description is enough to see what is wrong with this protocol. There is another type of transforming edge that produces a term of the same form as the incoming test component. This is the initiator's encrypting edge, in the case in which the initiator is B. Thus, the attacker can wait until B needs to authenticate itself to any responder, and can then execute the attack shown in Figure 22. Woo and Lam state that they assume that a principal can detect when it receives an encrypted unit that it has constructed itself; so perhaps this attack is not entirely "fair." See [3] for additional discussion.

Yet another problem (also discussed in [3]) exists. Even when the server constructs the term $\{|N_b|\}_{K_{BS}}$, this term does not fully determine the parameters to the server strand. A second attack on Woo-Lam exploits this. The attacker starts two sessions with the responder B. In one he purports to be A; in the other he uses some identity C he has somehow captured, so that $K_{CS} \in \mathsf{K}_{\mathcal{P}}$. He then switches the nonce N_b that B generates, intended to authenticate A, into the session with C, so that B sends $\{|C, \{|N_b|\}_{K_{CS}}|\}_{K_{BS}}$ to the server. The server then generates $\{|N_b|\}_{K_{BS}}$, which is the test component for B's session with A. The attacker then makes this appear to belong to that session. The auxiliary session with C fails to complete.

The Woo-Lam example is included here to illustrate how useful the authentication tests are as a heuristic used to find problems in protocols. They may be used for this purpose even in a case in which some of the official constraints on the authentication test are not satisfied. For instance, in the Woo-Lam protocol, the test component $\{|N_b|\}_{K_{BS}}$ could also occur as a proper subterm of a regular node, namely the message from a responder to the server. However, the authentication tests still model the reasoning of a protocol designer well enough to suggest where failures will lie.

6 Designing a Protocol: A Rational Reconstruction

The outgoing, incoming, and unsolicited tests, and the authentication results that apply to them, suggest a protocol design process. At our level of abstraction, authentication protocol design is largely a matter of selecting authentication tests, and constructing a unique regular transforming edge to satisfy each.[2] We will illustrate this process by an example, a possible rational reconstruction leading to the Needham-Schroeder-Lowe protocol.

It is important to start by deciding the goals to be achieved. Let us assume that we intend to construct a protocol in which the initiator A and responder B each generate a fresh, secret value, N_a and N_b respectively. They want to share these values between themselves without disclosing them to any other party. Each should learn that the other has proceeded far enough in the protocol to have received the values. Perhaps the principals intend to hash the two values together to produce a session key for an encrypted conversation. We will try to accomplish our goals without using excessive messages.

We must also stipulate the cryptographic conditions under which the protocol will operate. In our case, the relevant assumption is that each principal has an asymmetric key pair, and can reliably obtain the other's public key. Perhaps some public key infrastructure is already in place.

From the goal it follows that A can use an authentication test using N_a, while B can use an authentication test using N_b. Given the assumption that the principals hold each other's public keys, this can be an outgoing test. A can use a test component of the form $\{|\cdots N_a \cdots|\}_{K_B}$ assuming K_B^{-1} is uncompromised. Only B will be able to extract N_a from this encrypted form.

By contrast, an incoming test is not suitable. For instance, an incoming component of the form $\{|\cdots N_a \cdots|\}_{K_B^{-1}}$ would ensure that the transforming edge lies on a strand of principal B, but would sacrifice the secrecy of N_a. Similarly, an incoming component of the form $\{|\cdots N_a \cdots|\}_{K_A}$ would preserve secrecy, but would not ensure that the transforming edge lies on a regular strand, much less a strand of principal B. Nested encryption might yield a usable incoming test, but is more computationally demanding and more fragile.

The value A receives back in the outgoing test must be encrypted in a key whose inverse is uncompromised, presumably K_A, to preserve secrecy. In addition, the first term must contain A's name, as otherwise B does not know which public key to use for the return message. Thus, the first steps for A will be of the form

$$+\{|N_a\, A|\}_{K_B} \Rightarrow -\{|\cdots N_a \cdots|\}_{K_A} \Rightarrow \cdots$$

[2] Of course, at other levels of abstraction there are other issues, concerning how to negotiate cryptographic algorithms, how to evaluate whether cryptography has been used safely, how to format messages, how to distribute certificates, how to align key streams, and so on, that are not considered at the current level of abstraction.

A similar argument shows that B will use an outgoing test of the form:

$$\cdots \Rightarrow +\{|\cdots N_b \cdots|\}_{K_A} \Rightarrow -\{|\cdots N_b \cdots|\}_{K_B} \Rightarrow \cdots$$

We save a message by observing that B's outgoing message can be combined with A's incoming message. Hence, B's behavior can take the form:

$$\begin{array}{ll} - & \{|N_a\, A|\}_{K_B} \Rightarrow \\ & +\{|N_a\, N_b \cdots|\}_{K_A} \Rightarrow -\{|\cdots N_b \cdots|\}_{K_B} \Rightarrow \cdots \end{array}$$

If we try to be clever, we may guess that the presence of N_a will identify the run to A. In that case, we discard the ellipsis in B's outgoing message. Since there is no need to add anything to the third message or after it, we obtain the Needham-Schroeder protocol:

$$-\{|N_a\, A|\}_{K_B} \Rightarrow +\{|N_a\, N_b|\}_{K_A} \Rightarrow -\{|N_b|\}_{K_B}$$

A more systematic approach is to check whether the values contained in B's outgoing test component suffice to identify a unique initiator strand as the transforming edge for N_b. They do not, because B's identity is not determined. This establishes that we need a correction like Lowe's:

$$-\{|N_a\, A|\}_{K_B} \Rightarrow +\{|N_a\, N_b\, B|\}_{K_A} \Rightarrow -\{|N_b|\}_{K_B}$$

We have now selected the complete message structure for the protocol. We must now check that we have done so correctly. There are five questions that need to be answered:

1. Is the set of penetrable keys P disjoint from the decryption keys for outgoing components, and disjoint from the encryption keys for incoming and unsolicited components?

2. Is any test component a proper subterm of a component of term(n) for any regular node n?

3. Are there ever two types of transforming edge that transform the same outgoing component, or produce the same incoming component?

4. Do the parameters contained in the test components completely determine the data values contained in the desired authentication guarantee?

5. If a data value is intended to remain secret, is it always protected by at least one key K whose corresponding decryption key K^{-1} is not penetrable?

The first two questions must be answered affirmatively to apply Authentication Tests 1–3, which then entail that there exist matching regular transforming edges.

But must those regular transforming edges lie on the strands that we expect them to (Question 3)? A common cause of authentication failure arises when

there is also another edge that can transform the same value (e.g. Neuman-Stubblebine with re-authentication and Woo-Lam). Alternatively, we may know that a transforming edge of the kind desired is present, but it may not determine all of the parameters that we would like to agree on (Question 4). This was the reason for the failure of the original Needham-Schroeder protocol, and for the second Woo-Lam failure.

If the third and fourth questions are answered affirmatively, then the authentication goals of the protocol will have been met. Finally, question 5 assures that the protocol's secrecy goals will also be met.

Protocol designers need to be alert when Question 3 and Question 4 receive negative answers. Then there are *unintended services*, situations in which the protocol itself offers a transformation that can be abused by the penetrator. We recommend that protocol designers, even when working without any formal framework, ask themselves whether their protocols offer any unintended services to assist the penetrator in achieving what the protocol regards as establishing authentication. Unintended services are easy to recognize, and they are a strong clue where an attack on a protocol may lie.

Acknowledgments We are grateful to Sylvan Pinsky and Al Maneki for encouragement, support, and many technical discussions. We are grateful to Jonathan Herzog for suggesting that we develop these ideas from the germinal form they had in another paper. He and Lenore Zuck also helped us to improve the content of the paper.

References

[1] Michael Burrows, Martín Abadi, and Roger Needham. A logic of authentication. *Proceedings of the Royal Society*, Series A, 426(1871):233–271, December 1989. Also appeared as SRC Research Report 39 and, in a shortened form, in ACM Transactions on Computer Systems 8, 1 (February 1990), 18-36.

[2] I. Cervesato, N. A. Durgin, P. D. Lincoln, J. C. Mitchell, and A. Scedrov. A meta-notation for protocol analysis. In *Proceedings, 12th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, June 1999.

[3] John Clark and Jeremy Jacob. A survey of authentication protocol literature: Version 1.0. University of York, Department of Computer Science, November 1997.

[4] Edmund Clarke, Somesh Jha, and Will Marrero. Using state space exploration and a natural deduction style message derivation engine to verify security protocols. In *Proceedings, IFIP Working Conference on Programming Concepts and Methods (*PROCOMET*)*, 1998.

[5] T. Dierks and C. Allen. The TLS protocol. RFC 2246, January 1999.

[6] D. Dolev and A. Yao. On the security of public-key protocols. *IEEE Transactions on Information Theory*, 29:198–208, 1983.

[7] Antonio Durante, Riccardo Focardi, and Roberto Gorrieri. CVS: A compiler for the analysis of cryptographic protocols. In *12th Computer Security Foundations Workshop Proceedings*, pages 203–212. IEEE Computer Society Press, June 1999.

[8] Joshua D. Guttman and F. Javier Thayer Fábrega. Authentication tests. In *Proceedings, 2000 IEEE Symposium on Security and Privacy*. May, IEEE Computer Society Press, 2000.

[9] Joshua D. Guttman and F. Javier Thayer Fábrega. Protocol independence through disjoint encryption. In *Proceedings, 13th Computer Security Foundations Workshop*. IEEE Computer Society Press, July 2000.

[10] Tzonelih Hwang, Narn-Yoh Lee, Chuang-Ming Li, Ming-Yung Ko, and Yung-Hsiang Chen. Two attacks on Neuman-Stubblebine authentication protocols. *Information Processing Letters*, 53:103–107, 1995.

[11] Gavin Lowe. An attack on the Needham-Schroeder public key authentication protocol. *Information Processing Letters*, 56(3):131–136, November 1995.

[12] 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.

[13] Gavin Lowe. Casper: A compiler for the analysis of security protocols. In *10th Computer Security Foundations Workshop Proceedings*, pages 18–30. IEEE Computer Society Press, 1997.

[14] Gavin Lowe. A heirarchy of authentication specifications. In *10th Computer Security Foundations Workshop Proceedings*, pages 31–43. IEEE Computer Society Press, 1997.

[15] Will Marrero, Edmund Clarke, and Somesh Jha. A model checker for authentication protocols. In Cathy Meadows and Hilary Orman, editors, *Proceedings of the DIMACS Workshop on Design and Verification of Security Protocols*. DIMACS, Rutgers University, September 1997.

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

[17] B. Clifford Neuman and Stuart G. Stubblebine. A note on the use of timestamps as nonces. *Operating Systems Review*, 27(2):10–14, April 1993.

[18] D. Otway and O. Rees. Efficient and timely mutual authentication. *Operating Systems Review*, 21(1):8–10, January 1987.

[19] Lawrence C. Paulson. Proving properties of security protocols by induction. In *10th IEEE Computer Security Foundations Workshop*, pages 70–83. IEEE Computer Society Press, 1997.

[20] Dag Prawitz. *Natural Deduction: A Proof-Theoretic Study*. Almqvist and Wiksel, Stockholm, 1965.

[21] 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.

[22] M. Tatebayashi, N. Matsuzaki, and D. Newman. Key distribution protocol for digital mobile communication systems. In G. Brassard, editor, *Advances in Cryptology:* crypto *'89*, volume 435 of *Lecture Notes in Computer Science*, pages 324–331. Springer Verlag, 1990.

[23] F. Javier Thayer Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Mixed strand spaces. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, June 1999.

[24] F. Javier Thayer Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 7(2/3):191–230, 1999.

[25] T. Y. C. Woo and S. S. Lam. Authentication for distributed systems. *Computer*, 25(1):39–52, January 1992.

[26] T. Y. C. Woo and S. S. Lam. A lesson on authentication protocol design. *Operating Systems Review*, pages 24–37, 1994.

[27] Thomas Y. C. Woo and Simon S. Lam. Verifying authentication protocols: Methodology and example. In *Proc. Int. Conference on Network Protocols*, October 1993.

A Strands, Bundles, and the Penetrator

In this appendix, we will introduce the basic strand space notions to be used in the remainder of the paper. This material is derived from [24], with a few small changes. For instance, the penetrator strands of type T and F were unnecessary and have now been eliminated from Definition A.9.

A.1 Strand Spaces

Consider a set A, the elements of which are the possible messages that can be exchanged between principals in a protocol. We will refer to the elements of A as *terms*. We assume that a *subterm* relation is defined on A. $t_0 \sqsubset t_1$ means t_0 is a subterm of t_1. We constrain the set A further below in Section A.3, and define a subterm relation there.

In a protocol, principals can either send or receive terms. We represent transmission of a term as the occurrence of that term with positive sign, and reception of a term as its occurrence with negative sign.

Definition A.1 *A* signed term *is a pair* $\langle \sigma, a \rangle$ *with* $a \in \mathsf{A}$ *and* σ *one of the symbols* $+, -$. *We will write a signed term as* $+t$ *or* $-t$. $(\pm\mathsf{A})^*$ *is the set of finite sequences of signed terms. We will denote a typical element of* $(\pm\mathsf{A})^*$ *by* $\langle \langle \sigma_1, a_1 \rangle, \ldots, \langle \sigma_n, a_n \rangle \rangle$.

A strand space *over* A *is a set* Σ *together with a trace mapping* $\mathrm{tr} : \Sigma \to (\pm\mathsf{A})^*$.

By abuse of language, we will still treat signed terms as ordinary terms. For instance, we shall refer to subterms of signed terms. We will usually represent a strand space by its underlying set of strands Σ.

Definition A.2 Fix a strand space Σ.

1. A *node* is a pair $\langle s, i \rangle$, with $s \in \Sigma$ and i an integer satisying $1 \leq i \leq \mathrm{length}(\mathrm{tr}(s))$. The set of nodes is denoted by $\mathcal{N}$. We will say the node $\langle s, i \rangle$ belongs to the strand s. Clearly, every node belongs to a unique strand.

2. If $n = \langle s, i \rangle \in \mathcal{N}$ then $\mathrm{index}(n) = i$ and $\mathrm{strand}(n) = s$. Define $\mathrm{term}(n)$ to be $(\mathrm{tr}(s))_i$, i.e. the ith signed term in the trace of s. Similarly, $\mathrm{uns\_term}(n)$ is $((\mathrm{tr}(s))_i)_2$, i.e. the unsigned part of the ith signed term in the trace of s.

3. There is an edge $n_1 \rightarrow n_2$ if and only if $\mathrm{term}(n_1) = +a$ and $\mathrm{term}(n_2) = -a$ for some $a \in \mathsf{A}$. Intuitively, the edge means that node n_1 sends the message a, which is received by n_2, recording a potential causal link between those strands.

4. When $n_1 = \langle s, i \rangle$ and $n_2 = \langle s, i+1 \rangle$ are members of $\mathcal{N}$, there is an edge $n_1 \Rightarrow n_2$. Intuitively, the edge expresses that n_1 is an immediate causal predecessor of n_2 on the strand s. We write $n' \Rightarrow^+ n$ to mean that n' precedes n (not necessarily immediately) on the same strand.

5. An unsigned term t *occurs in* $n \in \mathcal{N}$ iff $t \sqsubset \mathrm{term}(n)$.

6. Suppose I is a set of unsigned terms. The node $n \in \mathcal{N}$ is an *entry point* for I iff $\mathrm{term}(n) = +t$ for some $t \in I$, and whenever $n' \Rightarrow^+ n$, $\mathrm{term}(n') \notin I$.

7. An unsigned term t *originates* on $n \in \mathcal{N}$ iff n is an entry point for the set $I = \{t' : t \sqsubset t'\}$.

8. An unsigned term t is *uniquely originating* iff t originates on a unique $n \in \mathcal{N}$.

If a term t originates uniquely in a particular strand space, then it can play the role of a nonce or session key in that structure.

$\mathcal{N}$ together with both sets of edges $n_1 \rightarrow n_2$ and $n_1 \Rightarrow n_2$ is a directed graph $\langle \mathcal{N}, (\rightarrow \cup \Rightarrow) \rangle$.

A.2 Bundles and Causal Precedence

A *bundle* is a finite subgraph of the graph $\langle \mathcal{N}, (\rightarrow \cup \Rightarrow) \rangle$, for which we can regard the edges as expressing the causal dependencies of the nodes.

Definition A.3 *Suppose $\rightarrow_{\mathcal{C}} \subset \rightarrow$; suppose $\Rightarrow_{\mathcal{C}} \subset \Rightarrow$; and suppose $\mathcal{C} = \langle \mathcal{N}_{\mathcal{C}}, (\rightarrow_{\mathcal{C}} \cup \Rightarrow_{\mathcal{C}}) \rangle$ is a subgraph of $\langle \mathcal{N}, (\rightarrow \cup \Rightarrow) \rangle$. $\mathcal{C}$ is a bundle if:*

1. *$\mathcal{N}_{\mathcal{C}}$ and $\rightarrow_{\mathcal{C}} \cup \Rightarrow_{\mathcal{C}}$ are finite.*

2. *If $n_2 \in \mathcal{N}_{\mathcal{C}}$ and $\mathrm{term}(n_2)$ is negative, then there is a unique n_1 such that $n_1 \rightarrow_{\mathcal{C}} n_2$.*

3. *If $n_2 \in \mathcal{N}_{\mathcal{C}}$ and $n_1 \Rightarrow n_2$ then $n_1 \Rightarrow_{\mathcal{C}} n_2$.*

4. *$\mathcal{C}$ is acyclic.*

In conditions 2 and 3, it follows that $n_1 \in \mathcal{N}_\mathcal{C}$, because $\mathcal{C}$ is a graph.

For our purposes, it does not matter whether communication is regarded as a synchronizing event or as an asynchronous activity. The definition of bundle formalizes a process communication model with three properties:

- A strand (process) may send and receive messages, but not both at the same time;
- When a strand receives a message t, there is a unique node transmitting t from which the message was immediately received;
- When a strand transmits a message t, many strands may immediately receive t.

Notational Convention A.4 *A node n is in a bundle $\mathcal{C} = \langle \mathcal{N}_\mathcal{C}, \rightarrow_\mathcal{C} \cup \Rightarrow_\mathcal{C} \rangle$, written $n \in \mathcal{C}$, if $n \in \mathcal{N}_\mathcal{C}$; a strand s is in $\mathcal{C}$ if all of its nodes are in $\mathcal{N}_\mathcal{C}$.*

If $\mathcal{C}$ is a bundle, then the $\mathcal{C}$-height *of a strand s is the largest i such that $\langle s, i \rangle \in \mathcal{C}$. $\mathcal{C}$-trace$(s) = \langle tr(s)(1), \ldots, tr(s)(m) \rangle$, where $m = \mathcal{C}$-height(s).*

Definition A.5 *If $\mathcal{S}$ is a set of edges, i.e. $\mathcal{S} \subset \rightarrow \cup \Rightarrow$, then $\prec_\mathcal{S}$ is the transitive closure of $\mathcal{S}$, and $\preceq_\mathcal{S}$ is the reflexive, transitive closure of $\mathcal{S}$.*

The relations $\prec_\mathcal{S}$ and $\preceq_\mathcal{S}$ are each subsets of $\mathcal{N}_\mathcal{S} \times \mathcal{N}_\mathcal{S}$, where $\mathcal{N}_\mathcal{S}$ is the set of nodes incident with any edge in $\mathcal{S}$.

Proposition A.6 *Suppose $\mathcal{C}$ is a bundle. Then $\preceq_\mathcal{C}$ is a partial order, i.e. a reflexive, antisymmetric, transitive relation. Every non-empty subset of the nodes in $\mathcal{C}$ has $\preceq_\mathcal{C}$-minimal members.*

We regard $\preceq_\mathcal{C}$ as expressing causal precedence, because $n \prec_\mathcal{S} n'$ holds only when n's occurrence causally contributes to the occurrence of n'. When a bundle $\mathcal{C}$ is understood, we will simply write $\preceq$. Similarly, "minimal" will mean $\preceq_\mathcal{C}$-minimal.

A.3 Terms, Encryption, and Freeness Assumptions

We will now specialize the set of terms A. In particular we will assume given:

- A set $\mathsf{T} \subseteq \mathsf{A}$ of texts (representing the atomic messages).
- A set $\mathsf{K} \subseteq \mathsf{A}$ of cryptographic keys disjoint from T, equipped with a unary operator $\mathbf{inv} : \mathsf{K} \to \mathsf{K}$. We assume that $\mathbf{inv}$ is an inverse mapping each member of a key pair for an asymmetric cryptosystem to the other, and each symmetric key to itself.
- Two binary operators $\mathbf{encr} : \mathsf{K} \times \mathsf{A} \to \mathsf{A}$ and $\mathbf{join} : \mathsf{A} \times \mathsf{A} \to \mathsf{A}$.

We follow custom and write $\mathbf{inv}(K)$ as K^{-1}, $\mathbf{encr}(K, m)$ as $\{\!|m|\!\}_K$, and $\mathbf{join}(a, b)$ as $a\,b$. If $\mathfrak{K}$ is a set of keys, $\mathfrak{K}^{-1}$ denotes the set of inverses of elements of $\mathfrak{K}$. We assume, like many others (e.g. [13, 15, 19]), that A is freely generated, which is crucial for the results in this paper.

Axiom 1 A *is freely generated from* T *and* K *by* **encr** *and* **join**.

Definition A.7 *The subterm relation* $\sqsubset$ *is defined inductively, as the smallest relation such that* $a \sqsubset a$; $a \sqsubset \{|g|\}_K$ *if* $a \sqsubset g$; *and* $a \sqsubset g\,h$ *if* $a \sqsubset g$ *or* $a \sqsubset h$.

By this definition, for $K \in \mathsf{K}$, we have $K \sqsubset \{|g|\}_K$ only if $K \sqsubset g$ already.

Definition A.8

1. *If* $\mathfrak{K} \subset \mathsf{K}$, *then* $t_0 \sqsubset_{\mathfrak{K}} t$ *if* t *is in the smallest set containing* t_0 *and closed under encryption with* $K \in \mathfrak{K}$ *and concatenation with arbitrary terms* t_1.
2. *A term* t_0 *is a* visible *subterm of* t *if* $t_0 \sqsubset_\emptyset t$.
3. *A term* t *is* simple *if it is not of the form* $g\,h$.
4. *A term* t_0 *is a* component *of* t *if* t_0 *is simple and* $t_0 \sqsubset_\emptyset t$.

We say that t_0 is a component of a node n if t_0 is a component of $\mathrm{term}(n)$.

A.4 Penetrator Strands

The atomic actions available to the penetrator are encoded in a set of *penetrator traces*. They summarize his ability to generate known messages, piece messages together, and apply cryptographic operations using keys that become available to him. A protocol attack typically requires hooking together several of these atomic actions.

The actions available to the penetrator are relative to the set of keys that the penetrator knows initially. We encode this in a parameter, the set of penetrator keys $\mathsf{K}_\mathcal{P}$.

Definition A.9 *A* penetrator trace *relative to* $\mathsf{K}_\mathcal{P}$ *is one of the following:*

M_t *Text message:* $\langle +t \rangle$ *where* $t \in \mathsf{T}$.

K_K *Key:* $\langle +K \rangle$ *where* $K \in \mathsf{K}_\mathcal{P}$.

$\mathsf{C}_{g,h}$ *Concatenation:* $\langle -g,\ -h,\ +g\,h \rangle$

$\mathsf{S}_{g,h}$ *Separation:* $\langle -g\,h,\ +g,\ +h \rangle$

$\mathsf{E}_{h,K}$ *Encryption:* $\langle -K,\ -h,\ +\{|h|\}_K \rangle$.

$\mathsf{D}_{h,K}$ *Decryption:* $\langle -K^{-1},\ -\{|h|\}_K,\ +h \rangle$.

$\mathcal{P}_\Sigma$ *is the set of all strands* $s \in \Sigma$ *such that* $\mathrm{tr}(s)$ *is a penetrator trace.*

A strand $s \in \Sigma$ *is a* penetrator strand *if it belongs to* $\mathcal{P}_\Sigma$, *and a node is a* penetrator node *if the strand it lies on is a penetrator strand. Otherwise we will call it a* non-penetrator *or* regular *strand or node. A node* n *is* M, K, *etc. node if* n *lies on a penetrator strand with a trace of kind* M, K, *etc.*

We assume that all strand spaces have an adequate supply of C, S, E, and D strands; by contrast, M and K strands vary, thus modeling the set of values the penetrator may know or be able to guess.