Disjoint Encryption and the Normal Penetrator

F. Javier THAYER Fábrega Joshua D. GUTTMAN

July 8, 1999

Abstract

In this paper we make two contributions. First, we use the formalism of multiprotocol strand spaces ([15], cf. [16]) to prove that two cryptographic protocols cannot interfere if they use encryption in non-overlapping ways. This theorem formalizes what has previously been a piece of folklore in the study of cryptographic protocols.

Our second contribution lies in the method that we use to prove this main theorem. We use a "penetrator normal form lemma" which shows that we may assume that penetrator decryption and separation always precede penetrator encryption and concatenation; this is akin to a result used by Clarke, Jha, and Marerro [4], and by Mitchell et al. [2]. However, combined with an assumption about the use of keys in the two protocols, it allows us to infer that certain paths from one protocol to the other (traversing penetrator actions) have a "subterm property," assuring that every node contains a common subterm of the terms at the first and last node of the path.

This method promises to simplify the analysis of single protocols also.

Contents

1 Introduction

Interference among protocols—or among sub-protocols of a single protocol with multiple cases—is a significant cause of protocol failure [3, 7, 8, 12, 15, 17].

However, there is a feeling, bolstered by some of the prudent advice of Abadi and Needham [1], that interference among protocols is a superficial problem, in the sense that it could be solved by administrative fiat in several ways.

For instance, suppose that keys were generated in such a way that two different protocols would never make use of the same key. Then how could they interfere?

Or alternatively, suppose that public keys and long term shared keys are shared among protocols, but that they are used in non-overlapping ways. Possibly each protocol (and sub-protocol) is assigned a distinct protocol identification number, and this protocol identification number is included in every ciphertext sent in the protocol. How then can a ciphertext for one protocol be misused in another protocol? If, in addition, session keys distributed in one protocol could never be used in the other, then it would appear impossible for the first protocol to undermine the security guarantees offered by the second.

In this paper, we will formalize these intuitions in a way that allows them to be proved correct. We say that two protocols have *disjoint encryption* if there is no ciphertext that could be used in runs of both. We say that a protocol is closed if keys distributed in runs of that protocol are used only in runs of that protocol. Our theorem (Proposition 5.10) is that a secondary protocol cannot undermine the security guarantees offered by a primary protocol if they have disjoint encryption, and the former is closed.

We use the strand space formalism [16, 15] as our vehicle. In this paper, we define a notion of equivalence for protocol executions or "bundles," as we call them (Definition 2.3) such that if a bundle $\mathcal{C}$ is a counterexample to a security guarantee for some primary protocol, then so is any equivalent bundle (Definitions 2.9, 2.10). Equivalent bundles may differ in the actions of the penetrator, or in runs of a secondary protocol.

We also examine the potential actions of the penetrator more carefully than it was necessary to do previously. We establish a normal form lemma for the penetrator (Proposition 4.6), which shows that we can always assume that the penetrator carries out actions on a particular order; this result is akin to one of Clarke, Jha and Marrero [4]. We show that given any bundle, there is an equivalent one in which the penetrator actions occur in the normalized order.

Using this normal form lemma and the assumption that the secondary protocol is closed, we show "subterm" properties that constrain certain messages constructed by a penetrator to be subterms of messages sent or received by the regular protocol participants (Propositions 5.6, 5.8). These form the main lemmas for establishing our theorem, that disjoint encryption and a closed secondary protocol suffice to ensure that the secondary protocol cannot interfere with the security guarantees of the primary protocol.

2 Strands, Bundles, and Equivalences

In this section, we will introduce the basic strand space notions to be used in the remainder of the paper. Sections 2.1–2.3 are derived from [16, 15]. Section 2.4 introduces a new notion, namely bundle equivalence.

2.1 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 2.3, 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 2.1 *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 2.2 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 $n_1 \to n_2$ 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 $t \sqsubset \text{term}(n)$.

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

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 $\langle \mathcal{N}, (\rightarrow \cup \Rightarrow) \rangle$.

2.2 Bundles and Causal Precedence

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

Definition 2.3 *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. $\mathcal{C}$ *is finite.*

2. *If* $n_2 \in \mathcal{N}_\mathcal{C}$ *and 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 2.4 *A node* n *is in a bundle* $\mathcal{C} = \langle \mathcal{N}_\mathcal{C}, \rightarrow_\mathcal{C}, \Rightarrow_\mathcal{C} \rangle$*, 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 2.5 *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 $\mathcal{S}$.

Lemma 2.6 *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.

2.3 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 **inv** is injective; that it maps each member of a key pair for an asymmetric cryptosystem to the other; and that it maps a symmetric key to itself.
- Two binary operators

$$\mathbf{encr} : \mathsf{K} \times \mathsf{A} \to \mathsf{A}$$
$$\mathbf{join} : \mathsf{A} \times \mathsf{A} \to \mathsf{A}$$

We will 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 k is a set of keys, k^{-1} denotes the set of inverses of elements of k.

The proofs in this paper use an assumption we will call the assumption of free encryption; many other authors (e.g. [9, 11, 13]) have made similar assumptions, dating back to Dolev and Yao [5], although not all have [6]. It stipulates that a ciphertext can be regarded as a ciphertext in just one way. For clarity of exposition we make a stronger assumption, namely that A is the algebra freely generated from T and K by the two operators **encr** and **join**, as embodied in Axiom 1.

Axiom 1 *For $m_0, m_0', m_1, m_1' \in \mathsf{A}$ and $K, K' \in \mathsf{K}$,*

1. $m_0\,m_1 = m_0'\,m_1' \Longrightarrow m_0 = m_0' \wedge m_1 = m_1'$
2. $\{m\}_K = \{m'\}_{K'} \Longrightarrow m = m' \wedge K = K'$
3. $m_0\,m_1 \neq \{m_0'\}_{K'}$

4. $m_0 m_1 \notin \mathsf{K} \cup \mathsf{T}$

5. $\{m_0\}_K \notin \mathsf{K} \cup \mathsf{T}$

Definition 2.7 *The subterm relation $\sqsubset$ is defined inductively, as the smallest relation such that:*

- *$a \sqsubset a$;*
- *$a \sqsubset \{g\}_K$ if $a \sqsubset g$;*
- *$a \sqsubset g\,h$ 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 2.8 1. *The* encryption-free *terms form the smallest set S including T and K and closed under concatenation.*

2. *A term t_0 is a* visible *subterm of t, written $t_0 \sqsubset_\emptyset t$, if t is in the smallest set containing t_0 and closed under concatenation with arbitrary terms t_1.*

A term t is encryption-free if and only if all of its subterms are visible.

2.4 Bundle Equivalences

Throughout the remainder of this paper, we will consider *multiprotocol strand spaces*. By this we mean a strand space in which the strands are partitioned into three sets. One set of strands is the *penetrator strands*, defined below (Definition 3.1). All strands other than penetrator strands are called *regular strands*. We further divide the regular strands into *primary strands* and *secondary strands*. The primary strands represent activities of principals in the protocol whose correctness we are investigating; secondary strands may be interwoven in the same bundle by means of actions of the penetrator, if the latter believes this will help to defeat some security goal of the primary protocol.

A node is a penetrator node or a regular node depending whether the strand it lies on is a penetrator or regular strand. A node is primary or secondary depending whether the strand it lies on is primary or secondary.

If Σ is a multiprotocol strand space, then we write $s \in \Sigma_1$ or $n \in \Sigma_1$ to indicate that the strand s or node n is primary. We write $s \in \Sigma_2$ or $n \in \Sigma_2$ to indicate that the strand s or node n is secondary.

Definition 2.9 *Bundles $\mathcal{C}, \mathcal{C}'$ are equivalent iff they have the same primary nodes.*

Naturally, this notion of equivalence is relative to the choice which strands to regard as primary and which to regard as secondary.

Definition 2.10 *A set ϕ of bundles is* invariant under bundle equivalences *if for every bundle equivalence $I : \mathcal{C} \to \mathcal{C}'$, $\mathcal{C} \in \phi \Rightarrow \mathcal{C}' \in \phi$.*

Observe that agreement and non-injective agreement properties [10, 16, 18] are invariant under bundle equivalences in this sense.

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

3 The Penetrator

3.1 Penetrator Strands

The atomic actions available to the penetrator are encoded in a set of *penetrator traces*. They summarize his ability to discard messages, generate well 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.

Definition 3.1 *A* penetrator trace *is one of the following:*

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

F_g**.** *Flushing:* $\langle -g \rangle$

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

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

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

$\mathsf{E}_{h,K}$**.** *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, F, etc. node if n lies on a penetrator strand with a trace of kind M, F, etc.

3.2 Public Values and Penetrator Actions

We regard a value as public in a strand space Σ if the penetrator can construct it without (unwitting) help from regular participants.

Definition 3.2 (Public, $P(\Sigma)$) *Let* Σ *be a multiprotocol strand space* Σ *with primary strands* Σ_1.

1. *A term t is* public *in Σ if there exists a bundle $\mathcal{C}$ in Σ and a node $n \in \mathcal{C}$ such that*
   (a) *$t = term(n)$, and*
   (b) *for all $n' \preceq_{\mathcal{C}} n$, n' is a penetrator node.*
2. *Let $P(\Sigma)$ be the least set containing:*
   (a) *K, if $K \in \mathsf{K}_{\mathcal{P}}$;*
   (b) *t, if $t \in \mathsf{T}$ and t does not originate uniquely on some $s \in \Sigma_1$,*

   and closed under:

   (c) *If $t, t' \in P$, then their concatenation $t\,t' \in P$;*
   (d) *If $t, K \in P$, then $\{t\}_K \in P$.*

By induction on the precedence relation $\preceq_{\mathcal{C}}$, we may infer:

Proposition 3.3 *Let Σ be a multiprotocol strand space with primary strands Σ_1. If t is public in Σ, then $t \in P(\Sigma)$.*

We cannot prove the converse, that $t \in P(\Sigma)$ implies that t is public, because Σ may lack strands that would be permitted under Definition 3.1. Definition 3.1 does not stipulate that a strand space has one or more strands with each possible penetrator trace. In particular, if Σ had strands s with $\mathrm{tr}(s) = \mathsf{M}_t$ for every $t \in \mathsf{T}$, then Σ would contain no nonces that could originate uniquely on primary strands. Similarly, we want to ensure that *keys* may be fresh in the sense that they originate only on primary strands. We also want to be able to ensure that keys are not disclosed, for instance long-term keys that are used but never uttered on primary strands. Such keys (if not originally known to an attacker) should never originate as a subterm of any term in Σ.

We say that Σ is full if these are the only strands lacking from $\mathcal{P}_\Sigma$.

Definition 3.4 *Σ is full if*

1. *$K \in \mathsf{K}_{\mathcal{P}}$ whenever K originates on any node $n \in \Sigma_2$, and*
2. *for every penetrator trace $\vec{a}$, either*
   (a) *$\vec{a} = \langle +t \rangle$ and $t \in \mathsf{T}$ originates uniquely on a primary strand $s' \in \Sigma_1$; or else*
   (b) *there exist countably many strands $s \in \Sigma$ such that* $\mathrm{tr}(s) = \vec{a}$.

The meaning of "full" depends crucially on which strands are designated as the primary strands.

If Σ is full, then by induction on the definition of $P(\Sigma)$, we can build "witnessing" bundles $\mathcal{C}$ in which the penetrator constructs each $t \in P(\Sigma)$. If $t \in \mathsf{K}$, then $\mathcal{C}$ consists of a single K_t-strand; if $k \in \mathsf{T}$, then $\mathcal{C}$ consists of a single M_t-strand. For compound terms t, E- and C-strands are added to build up t. Hence:

Proposition 3.5 *Let Σ be a full multiprotocol strand space. If $t \in P(\Sigma)$, then t is public in Σ.*

Because there are countably many copies of each penetrator strand, while bundles are finite, in a full strand space we can always assume that a "witnessing" bundle $\mathcal{C}_0$ is disjoint from any other bundle $\mathcal{C}$ in which we may be interested.

Proposition 3.6 *Suppose that Σ is full, t is public in Σ, and $\mathcal{C}$ is a bundle in Σ. There exists a bundle $\mathcal{C}'$ equivalent to $\mathcal{C}$ such that there exists a node $n \in \mathcal{C}$ such that:*

1. *$term(n) = t$; and*
2. *$n' \preceq_{\mathcal{C}'} n$ implies n' is a penetrator node and $n' \notin \mathcal{C}$.*

Proof Let $\mathcal{C}_0$ be disjoint from $\mathcal{C}$ and witness for t being public. Let $\mathcal{C}'$ be the graph whose nodes are those in either $\mathcal{C}$ or $\mathcal{C}_0$, and whose edges are those in either $\mathcal{C}$ or $\mathcal{C}_0$. Observe that $\mathcal{C}'$ is a bundle. Moreover, $\mathcal{C}_0$ contains the required node n.

Proposition 3.7 *Suppose that $\mathcal{C}$ is a bundle and N is a set of nodes. Let G be the graph such that*

1. *$m \in G$ if $m \in \mathcal{C}$ and $m \preceq_{\mathcal{C}} n$ for some $n \in N$;*
2. *$m_1 \Rightarrow m_2$ if $m_1, m_2 \in G$ and $m_1 \Rightarrow m_2$ in $\mathcal{C}$;*
3. *$m_1 \to m_2$ or $m_1, m_2 \in G$ and $m_1 \to m_2$ in $\mathcal{C}$.*

Then G is a bundle, called $\mathcal{C} \mid N$.

3.3 Redundancies

Definition 3.8 *A* redundancy *in a bundle $\mathcal{C}$ is any labeled subgraph of $\mathcal{C}$ that has one of the forms given in figures 1-2.*

Each redundancy consists of

1. Nodes on two penetrator strands, indicated by the symbol $\bullet$ and a number of "fringe" nodes indicated by the symbol $\circ$,
2. Inward edges $\circ \to \bullet$, outward edges $\bullet \to \circ$ and internal edges $\bullet \to \bullet$.

Note that some, any or all of the nodes indicated by $\circ$ in the redundancy diagrams can be regular nodes. Note also that any of the positive nodes of the diagram may have outgoing arrows not indicated in the figure.

The presence of redundancies in a bundle makes it much 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.

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

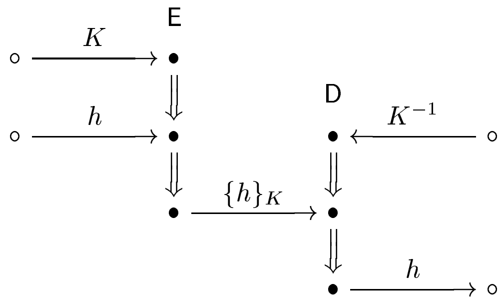

Figure 1: E-D Redundancy

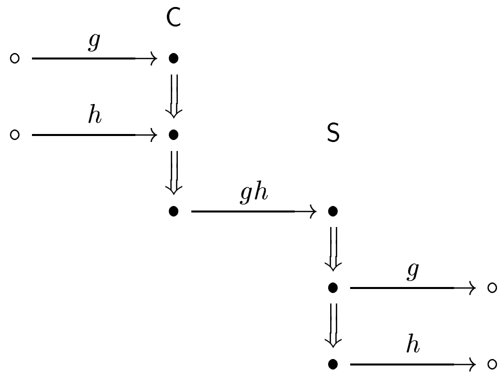

Figure 2: C-S Redundancy

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 3.9 *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 path $n = n_1 \mapsto \cdots \mapsto n_k = n'$ 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 path in $\mathcal{C}'$ is a subsequence of the nodes of a path 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. ■

Proposition 3.10 *Suppose $\mathcal{C}$ is a bundle and $\mathcal{C}'$ is obtained from $\mathcal{C}$ by a graph operation such that, for some n, m, and m',*

1. *A single edge $m \to n$ is deleted;*
2. *A single edge $m' \to n$ is added;*
3. *It is not the case that $n \preceq_{\mathcal{C}} m'$.*

Then $\mathcal{C}'$ is equivalent to $\mathcal{C}$.

Proof. If $\mathcal{C}'$ contains a cycle but $\mathcal{C}$ does not, then the cycle must traverse the new edge $m' \to n$. Since this is the only edge in $\mathcal{C}'$ but not $\mathcal{C}$, the remainder of the cycle must exist in $\mathcal{C}$. But that is a path p of the form $n = n_1 \mapsto \cdots \mapsto n_k = m'$ in $\mathcal{C}$, so $n \preceq_{\mathcal{C}} m'$.

There can be no gregarious or lonely negative node in $\mathcal{C}'$, as every negative node in $\mathcal{C}'$ has the same number of in-arrows as in $\mathcal{C}$. Again, no strands have been added. ■

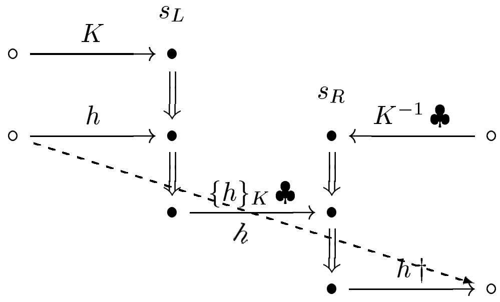

Figure 3: E-D Redundancy Elimination

Proposition 3.11 *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 $\prec_{\mathcal{C}}$.*

PROOF. Consider each one of the redundancy types shown in figures 3-4. 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. For each such subgraph,

1. Add the edges indicated by the dotted lines in the figure. 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 $n \rightarrow n'$, 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 $m \rightarrow n'$ going out of s_R are removed as well. The deleted edges are indicated by the symbol †. Note that in step 1, we have added an arrow into n' so that removal of $m \rightarrow n'$ 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 3.9. Hence, the resulting graph is bundle equivalent to $\mathcal{C}$, and its ordering is a restriction of the ordering of $\mathcal{C}$. ■

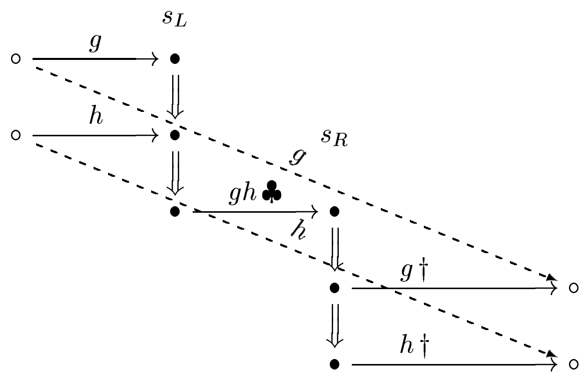

Figure 4: C-S Redundancy Elimination

4 Paths

$m \Rightarrow^+ n$ means that m, n are nodes on the same strand with n occuring after m (Definition 2.2, Clause 4). The notation $m \longmapsto n$ means:

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

A *path* p through $\mathcal{C}$ is any finite sequence of nodes and edges $n_1 \longmapsto n_2 \longmapsto \cdots \longmapsto n_k$. A segment of the path p is any path $n_i \longmapsto n_{i+1} \longmapsto \cdots \longmapsto n_j$ where $1 \leq i \leq j \leq k$. We assume all paths begin on a positive node. Clearly, $n \prec_{\mathcal{C}} n'$ whenever there is a path $n = n_1 \longmapsto n_2 \longmapsto \cdots \longmapsto n_k = n'$.

Definition 4.1 *A $\Rightarrow^+$ edge s_1 immediately precedes an edge s_2 iff they are separated by a single $\rightarrow$ edge.*

4.1 Penetrator Paths and Normal Bundles

A penetrator path is one in which all nodes other than the first or the last node are penetrator nodes.

Definition 4.2 *Consider a $\Rightarrow^+$-edge through penetrator nodes. There are four penetrator strand types with a negative node followed by a positive node. Hence, we say:*

1. *The edge is* constructive *if it is part of a* E *or* C *strand;*
2. *The edge is* destructive *if it is part of a* D *or if it is part of a* S *strand.*

We also say of a penetrator node n:

1. *n is* final *if it an* F *node.*

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

Proposition 4.3 *If a constructive edge P is immediately followed by a destructive edge Q then either there are h, K such that P is part of a $\mathsf{E}_{h,K}$ strand and Q is part of a $\mathsf{D}_{h,K}$ strand or there are g, h such that P is part of a $\mathsf{C}_{g,h}$ strand Q is part of a $\mathsf{S}_{g,h}$ strand.*

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

Proposition 4.4 *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 4.3, 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 the Prawitz's notion of normal derivation [14]:

Definition 4.5 *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] independently observed this analogy between penetrator activities and natural deduction inferences. By Propositions 3.11 and 4.4, we may infer:

Proposition 4.6 (Penetrator Normal Form Lemma) *For any bundle $\mathcal{C}$ there exists an equivalent normal bundle $\mathcal{C}'$.*

4.2 Descending and Ascending Paths

Definition 4.7 *A penetrator path is* descending *if for all adjacent nodes $n \longmapsto n'$ on the path $term(n') \sqsubset term(n)$.*

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

Clearly, any descending path cannot contain constructive edges, and any ascending path cannot contain destructive edges. However, a path containing only destructive edges may not be descending, since a destructive path may traverse a decryption strand entering through the key transmission edge (Figure 5). Call the edge labeled K^{-1} in Figure 5 a D-key edge. The other edge into 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 6). Call the edge labeled K in Figure 6 a E-key edge. The other edge into E strand is a E-plaintext edge.

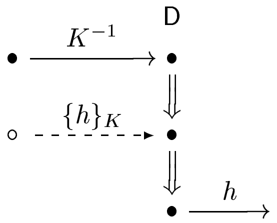

Figure 5: Entering a D strand through a key edge

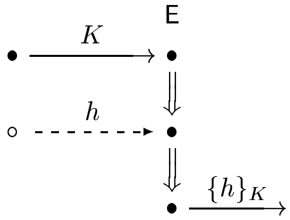

Figure 6: Entering a E strand through a key edge

Proposition 4.8 *A destructive path which enters decryption strands only through* D*-cyphertext edges is descending.*

A constructive path which enters encryption strands only through E*-plaintext edges is ascending.*

However, this symmetry is superficial. K is an atomic term, having no proper subterms. Hence, when an constructive path enters a E strand through a key edge, this can only be the first node on the path. By contrast, a destructive path may enter D strands through key edges repeatedly. We choose to focus on destructive paths in this section; we will return to constructive paths in Proposition 5.8.

Definition 4.9 *If n is a node, $\bar{n}$ is the set of regular nodes n' for which there exist K, h such that $K^{-1} \sqsubset term(n)$ and $\{h\}_K \sqsubset term(n')$.*

If n is a node, $\tilde{n}$ is the set of regular nodes n' for which there exist K, h such that $K \sqsubset term(n)$ and $\{h\}_K \sqsubset term(n')$.

A set S of nodes is bar-closed *iff for every $n \in S$, $\bar{n} \subset S$. It is* tilde-closed *iff for every $n \in S$, $\tilde{n} \subset S$.*

Clearly, the set of regular nodes is both bar-closed and tilde-closed.

Proposition 4.10 (Path Correction Lemma) *Suppose $S \subset \mathcal{C}$ is bar-closed. Let $p = n_1 \longmapsto n_2 \longmapsto \cdots \longmapsto n_k$ be a destructive penetrator path starting at $n_1 \in S$. Then there is a descending penetrator path in $\mathcal{C}$ of the form $n'_1 \longmapsto n'_2 \longmapsto \cdots \longmapsto n'_{k'}$ such that $n'_{k'} = n_k$ and $n'_1 \in S$.*

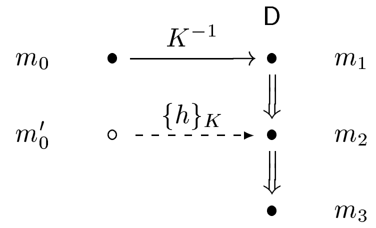

Figure 7: First D-key Edge in Path p

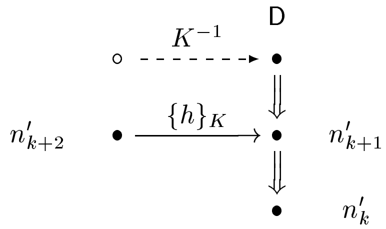

Figure 8: Prefixing cyphertext transmission edge

Proof. If p does not contain any D-key edges, then by Proposition 4.8, p is a descending penetrator path. Otherwise, let $m_0 \rightarrow m_1$ be the first D-key edge of p as illustrated in Figure 7. Thus the segment of p preceding the node m_0 is descending, and in particular, $K^{-1} \sqsubset \text{term}(n_1)$. If $m_1 \Rightarrow^+ m_3$ is not an edge of the original path p, then p terminates at the node m_1. In this case, p is clearly a descending path. Otherwise, $m_0 \rightarrow m_1 \Rightarrow^+ m_3$ is a segment of p. Define inductively a new penetrator path $\{n_{k+1}' \longmapsto n_k'\}_k$ with the following properties:

1. $n_0' = m_3$
2. $\{n_{i+1}' \longmapsto n_i'\}_i$ is either infinite or its initial node n_k' is a regular node.
3. $\{n_i'\}_i$ is a descending path.

The construction of the new path proceeds from the end of the path towards the front. Suppose

$$n_k' \longmapsto n_{k-1}' \longmapsto \cdots \longmapsto n_0' = m_3 \tag{1}$$

is a descending penetrator path. If n_k' is a regular node, the construction is finished. Otherwise, n_k' is on a penetrator strand s. Let us consider two cases:

s is a D strand. In this case, extend the path backwards by following the cyphertext transmission edge into s. Thus, referring to Figure 8, prefix the segment $n_{k+2}' \rightarrow n_{k+1}' \Rightarrow n_k'$ to the original path 1 . It is clear that the extended path $n_{k+2}' \rightarrow n_{k+1}' \Rightarrow n_k' \longmapsto \cdots n_0'$ is also descending.

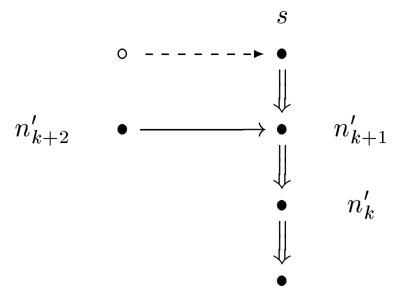

Figure 9: Extension for non-D Strands

s is not a D strand. Since $\{h\}_K \sqsubset n'_k$, s cannot be an M or K strand. Thus there must exist at least one $\rightarrow$ edge in $\mathcal{C}$ pointing into s. In other words, the stand s must be situated within $\mathcal{C}$ as depicted in figure 9.

In this case, extend the original path by prefixing the segment $n'_{k+2} \rightarrow n'_{k+1} \Rightarrow^+ n'_k$ to it. By the fact $\mathcal{C}$ has no redundancies, the prefixed segment must be a destructive segment. This means s is a strand of type S. It follows that in this case also, the extended path is descending.

Now by finiteness and acyclicity, the resulting sequence of edges $\{n'_k \longmapsto n'_{k-1}\}_k$ cannot be infinite. It therefore must be finite and by construction, the first node of the path must be a regular node. It follows there exists a descending penetrator path $n'_k \longmapsto n'_{k-1} \longmapsto \cdots \longmapsto m_3$ with n'_k a regular node. Now $\{h\}_k \sqsubset \text{term}(n'_k)$ and $K^{-1} \sqsubset \text{term}(n_1)$. Since S is bar-closed, it follows $n'_k \in S$.

With this construction, we successively remove all of the decryption key edges of the path. ■

4.3 Bridges

Definition 4.11 *A* bridge *in a bundle $\mathcal{C}$ is a message transmission edge $m \rightarrow n$ embedded in a subgraph of one the types shown in Figures 10–13.*

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

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

Note that any edge between regular nodes is a external bridge. The source m of a bridge $m \rightarrow n$ is never on a constructive penetrator strand, and the target n is never on a destructive penetrator strand.

Proposition 4.12 *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 $\mathcal{C}'$.*

$$
\begin{tikzcd}
\text{Regular} & \text{Constructive} \\
\circ \arrow[r, "h"] & \bullet \arrow[d, Rightarrow] \\
& \bullet
\end{tikzcd}
$$

Figure 10: Entry Bridge

$$
\begin{tikzcd}
\text{Destructive} & \text{Regular} \\
\bullet \arrow[d, Rightarrow] & \\
\bullet \arrow[r, "h"] & \circ
\end{tikzcd}
$$

Figure 11: Exit Bridge

$$
\begin{tikzcd}
\text{Regular} & \text{Regular} \\
\circ \arrow[r, "h"] & \circ
\end{tikzcd}
$$

Figure 12: External Bridge

$$
\begin{tikzcd}
\text{Destructive} & \text{Constructive} \\
\bullet \arrow[d, Rightarrow] & \\
\bullet \arrow[r, "h"] & \bullet \arrow[d, Rightarrow] \\
& \bullet
\end{tikzcd}
$$

Figure 13: Internal Bridge

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 as shown below:

$$\begin{tikzcd}
\circ \arrow[r, "g\,h"] & \bullet \arrow[d, Rightarrow] & & \\
& \bullet \arrow[r, "g"] \arrow[d, Rightarrow] & \bullet \arrow[d, Rightarrow] & \\
& \bullet \arrow[r, "h"] & \bullet \arrow[d, Rightarrow] & \\
& & \bullet \arrow[r, "g\,h"] & \circ
\end{tikzcd}$$

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. ■

Definition 4.13 *A bridge $m \to n$ is* minimal *iff there is no node $m_1 \prec m$ for which term(m_1) = term(m).*

Proposition 4.14 *Given any bundle $\mathcal{C}$ there exists an equivalent bundle $\mathcal{C}'$ with the same nodes as $\mathcal{C}$ and with only minimal bridges. Moreover, the ordering of $\mathcal{C}'$ is a restriction of the ordering of $\mathcal{C}$. If $\mathcal{C}$ is normal, $\mathcal{C}'$ can be taken to be normal as well.*

Proof. Suppose $m \to n$ is a bridge. Suppose there is a node $m_0 \prec m$ with $\mathrm{term}(m_0) = \mathrm{term}(m)$. We can assume m_0 is a minimal node with this property. By minimality, m_0 must be a positive node. Delete the edge $m \to n$ and replace it with an edge $m_0 \to n$. The new graph $\mathcal{C}'$ is clearly a bundle with a weaker ordering. No new bridges are added by this operation other than possibly $m_0 \to n$. But this new bridge is clearly minimal in $\mathcal{C}'$, since the ordering of $\mathcal{C}'$ is weaker than that of $\mathcal{C}$.

If the original bundle is normal, then we want to ensure that $\mathcal{C}'$ is also normal. However, $\mathcal{C}'$ can contain a redundancy only if n is on a destructive penetrator strand. This, however, is ruled out since $m \to n$ is a bridge in $\mathcal{C}$, and the target of a bridge is never a destructive strand. ■

5 Protocol Independence

We turn now to proving our main theorem using the machinery we have developed.

We will show that if, in a multiprotocol strand space Σ, the primary and secondary protocols use encryption in non-overlapping ways, and if a bundle $\mathcal{C}$ is a counterexample to some secrecy or authentication property in Σ, then there is a counterexample $\mathcal{C}'$ in which no secondary nodes occur. Hence, the correctness of the primary protocol is independent of the presence of the secondary protocol.

Recall from Definition 4.9 that a set S of nodes is bar-closed if, whenever $n \in S$, $K^{-1} \sqsubset n$ and $\{h\}_K \sqsubset n'$ with n' regular, then $n' \in S$. It is tilde-closed if, whenever $n \in S$, $K \sqsubset n$ and $\{h\}_K \sqsubset n'$ with n' regular, then $n' \in S$.

Definition 5.1 (Closed, Disjoint Encryption)

1. *Σ is* closed *if Σ_2 is both tilde-closed and bar-closed.*
2. *Σ has* disjoint encryption *if there is no encrypted term $\{h\}_K$ such that $\{h\}_K \sqsubset n$ and $\{h\}_K \sqsubset n'$ where $n \in \Sigma_1$ and $n' \in \Sigma_2$.*

Our intuitive phrase

> the primary and secondary protocols in Σ use encryption in non-overlapping ways

will be formalized as the assumption that Σ is closed and has disjoint encryption.

Definition 5.2 (Linking Paths) *$p = n_1 \longmapsto n_2 \longmapsto \cdots \longmapsto n_k$ is a* linking path *if it is a penetrator path and either*

1. *$n_1 \in \Sigma_1$ and $n_k \in \Sigma_2$, or*
2. *$n_1 \in \Sigma_2$ and $n_k \in \Sigma_1$.*

In case 1, the linking path is out-bound; in case 2, the linking path is in-bound.

A linking path never contains an initial penetrator node (an M or K node) or a final penetrator node (an F node). All penetrator nodes on a linking path must thus belong to constructive or destructive $\Rightarrow$-segments.

Proposition 5.3 *Let $\mathcal{C}$ be a normal bundle. Every linking path $p = n_1 \longmapsto n_2 \longmapsto \cdots \longmapsto n_k$ in $\mathcal{C}$ traverses exactly one bridge $n_j \to n_{j+1}$.*

If p is an in-bound linking path, then $n_{j+1} \notin \Sigma_2$, and if p is an out-bound linking path, then $n_j \notin \Sigma_2$.

PROOF. There are several cases depending on the form of p. First, if $k = 2$, so that $n_1 \in \Sigma_1$ and $n_2 \in \Sigma_2$ or vice versa, then $n_1 \longmapsto n_2$ is itself an external bridge. There is no other edge on the path, hence no other bridge. Moreover, $n_2 \notin \Sigma_2$ if the path is in-bound, and $n_1 \notin \Sigma_2$ if it is out-bound.

Otherwise, there is at least one penetrator $\Rightarrow$-segment on p. Since p is a linking path, only the first and last nodes are regular. Hence there is no external bridge on p. There are now three possibilities:

1. There are only destructive penetrator $\Rightarrow$-segments on p. Hence, $n_{k-1} \longmapsto n_k$ is an exit bridge, and there can be no entry bridge or internal bridge. Moreover, $n_{k-1} \notin \Sigma_2$; if p is in-bound, $n_k \notin \Sigma_2$.

2. There are only constructive penetrator $\Rightarrow$-segments on p. Hence, $n_1 \longmapsto n_2$ is an entry bridge, and there can be no exit bridge or internal bridge.

   Moreover, $n_2 \notin \Sigma_2$; if p is out-bound, $n_1 \notin \Sigma_2$.

3. There are both destructive and constructive penetrator $\Rightarrow$-segments on p. Since $\mathcal{C}$ is normal, every destructive segment precedes every constructive segment. Hence, there can be no entry bridge or exit bridge, and there must be one internal bridge joining the last destructive $\Rightarrow$-segment to the first constructive $\Rightarrow$-segment.

   Moreover, both nodes on an internal bridge are penetrator nodes. ■

Definition 5.4 *Let $\mathcal{C}$ be a normal bundle and p a linking path in $\mathcal{C}$.*

The bridge *of p is the unique edge $n_j \rightarrow n_{j+1}$ in p that is a bridge. The* path bridge term *of p, written $pbt(p)$, is the bridge term of the bridge of p.*

We will first study in-bound linking paths, after which we will consider out-bound linking paths.

5.1 In-Bound Linking Paths

Proposition 5.5 *Suppose Σ is closed and $\mathcal{C}$ is a normal bundle. Let $p' = n_j \rightarrow n_{j+1} \longmapsto \cdots \longmapsto n_k$ be the bridge and (possibly null) constructive portion of an in-bound linking path p in $\mathcal{C}$. Then $n_j \rightarrow n_{j+1}$ is not the key edge of an E strand.*

Therefore, $term(n_j) \sqsubset term(n_k)$.

Proof. Since $\mathcal{C}$ is normal, either $n_j \in \Sigma_2$ or n_j is the last node of the destructive portion of p. In the second case, since Σ is closed, the path correction lemma 4.10 entails that $\mathrm{term}(n_j) \sqsubset \mathrm{term}(n'_1)$ for some $n'_1 \in \Sigma_2$. Hence in either case $\mathrm{term}(n_j)$ is a subterm of the term at some secondary node n'_1.

If n_j is the key node of an E strand, then $\mathrm{term}(n_j) = K$ and $\mathrm{term}(n_{j+1})$ is $\{h\}_K$ for some h, K. Moreover, the remainder of this path is constructive, and certainly cannot enter any E strand through the key node. Hence $\{h\}_K \sqsubset \mathrm{term}(n_k) \in \Sigma_1$, while $\mathrm{term}(n_j) = K \sqsubset \mathrm{term}(n'_1)$, which is in Σ_2, contradicting the assumption that Σ_2 is tilde-closed. ■

Proposition 5.6 *Suppose Σ is closed, and $\mathcal{C}$ is a normal bundle. Let $p = n_1 \longmapsto \cdots \longmapsto n_k$ be an in-bound linking path in $\mathcal{C}$, and let $n_j \rightarrow n_{j+1}$ be the bridge of p. Then:*

1. *$pbt(p) \sqsubset term(n'_1)$ for some $n'_1 \in \Sigma_2$ such that $n'_1 \prec n_j$, and*

2. *$pbt(p) \sqsubset term(n_k)$, where $n_k \in \Sigma_1$.*

Proof. Apply the Path Correction Lemma 4.10 and Proposition 5.5. ■

Proposition 5.7 *Suppose Σ is closed and has disjoint encryption, and suppose $\mathcal{C}$ is a normal bundle in Σ. If p is an in-bound linking path in $\mathcal{C}$, then $pbt(p)$ is encryption-free.*

PROOF. By Proposition 5.6,

$$\text{pbt}(p) \sqsubset \text{term}(n) \quad \text{and} \quad \text{pbt}(p) \sqsubset \text{term}(n')$$

where $n \in \Sigma_1$ and $n' \in \Sigma_2$. Hence, by the definition of disjoint encryption, $\text{pbt}(p)$ is encryption-free. ∎

5.2 Out-Bound Linking Paths

Proposition 5.8 *Suppose that Σ is any multiprotocol strand space, and let $a \in \mathsf{K} \cup \mathsf{T}$ originate uniquely on a primary node $m_1 \in \Sigma_1$. Let $\mathcal{C}$ be a bundle in Σ such that there exists a secondary node $m_2 \in \mathcal{C}$ with $a \sqsubset term(m_2)$.*

There exists an out-bound linking path $p = n_k \longmapsto \cdots \longmapsto n_1$ such that $m_1 \preceq n_k$, $n_1 \preceq m_2$, and $a \sqsubset term(n_j)$ for all $j = k$ to 1.

If in addition Σ has disjoint encryption, then $pbt(p)$ is encryption-free.

PROOF. Consider the set $S = \{n \in \mathcal{C} : a \sqsubset \text{term}(n) \wedge n \preceq m_2 \wedge n \in \Sigma_2\}$. S is non-empty, because $m_2 \in S$; let n_1 be a minimal member of S. Define inductively a new penetrator path $\{n_j\}_j$ starting at n_1 with the following properties:

1. $\{n_j\}_j$ is either infinite or its first node n_k is a primary node;
2. $a \sqsubset \text{term}(n_j)$ for all $n_j \in \{n_j\}_j$;
3. n_j is a penetrator node whenever $j \neq 1$ and $j \neq k$.

The construction of the new path proceeds from the end of the path towards the front. Suppose

$$n_j \longmapsto n_{j-1} \longmapsto \cdots \longmapsto n_1 \tag{2}$$

If n_j is a negative node, then there is a unique positive node n_{j+1} such that $n_{j+1} \rightarrow n_j$ in $\mathcal{C}$ and $\text{term}(n_{j+1}) = \text{term}(n_j)$. Since $a \sqsubset \text{term}(n_{j+1})$ and $n_{j+1} \preceq n_1$, the minimality of n_1 ensures that $n_{j+1} \notin \Sigma_2$. This case applies at the first step.

If n_j is a positive penetrator node, then there must be an earlier negative node n_{j+1} on the same strand such that $a \sqsubset \text{term}(n_{j+1})$, for otherwise a would originate on a penetrator strand, contrary to assumption.

Since $\mathcal{C}$ is finite and acyclic, while $n_i \prec n_j$ when $i > j$, it is impossible for $\{n_j\}_j$ to be infinite. Therefore it terminates with a primary node n_k. Moreover $m_1 \preceq n_k$ by Lemma 2.9 of [16].

Suppose now that Σ has disjoint encryption. If p traverses the key node of a E-strand with key K, then since $a \sqsubset K$, we may infer $a = K$, whence a occurs free of encryption at this node. Likewise, if p traverses the key node of a D-strand with key K^{-1}, then since $a \sqsubset K^{-1}$, we may infer $a = K^{-1}$, whence a occurs free of encryption at this node.

If p never traverses the key node of a E-strand or D-strand, then by Proposition 4.8 the destructive portion of p is descending and the constructive portion of p is ascending. Letting $n_{j+1} \rightarrow n_j$ be the bridge of p, we may infer that $\mathrm{term}(n_{j+1}) \sqsubset \mathrm{term}(n_k)$, where $n_k \in \Sigma_1$ while $\mathrm{term}(n_j) \sqsubset \mathrm{term}(n_1)$, where $n_1 \in \Sigma_2$. Hence, since Σ has disjoint encryption, $\mathrm{term}(n_{j+1}) = \mathrm{term}(n_j)$ is encryption-free. ■

Proposition 5.9 *Suppose that Σ has disjoint encryption, and $\mathcal{C}$ is a normal bundle with simple, minimal bridges.*

If $a \in \mathsf{T} \cup \mathsf{K}$ originates uniquely on a primary node of $\mathcal{C}$, then there is no in-bound linking path p with $pbt(p) = a$.

PROOF. Suppose that p is an in-bound linking path with $\mathrm{pbt}(p) = a$. By Proposition 5.6, there is a node $n_1' \in \Sigma_2$ such that n_1' precedes the bridge of p in $\mathcal{C}$ and $a \sqsubset \mathrm{term}(n_1')$.

By Proposition 5.8, there exists an out-bound linking path p' such that:

1. The last node n_k' of p' precedes the first node n_1 of p;
2. $a \sqsubset \mathrm{term}(n)$ for every node n on p'; and
3. $\mathrm{pbt}(p')$ is encryption-free.

Since $\mathcal{C}$ has simple bridges, $\mathrm{pbt}(p')$ is also not concatenated. So $\mathrm{pbt}(p') = a$.

But since $n_k' \prec n_1$, the bridge of p' precedes the bridge of p. Since they have the same term, this contradicts the assumption that $\mathcal{C}$ has minimal bridges. ■

This proposition establishes, in effect, that the penetrator derives no benefit from out-bound linking paths. Any processing that the secondary protocol provides, in generating some complex term t containing a, must simply be undone on the corresponding in-bound linking path before a is returned to a primary strand. It is easier not to use the secondary strands at all, but just to rebuild whatever value uses a, and is consumed by a primary strand, by directly connecting the bridge of p' to the constructive portion of p.

5.3 The Protocol Independence Theorem

We now prove that when the protocols of the mixed strand space use encryption in non-overlapping ways, then for any counterexample to a property invariant under bundle equivalences—in particular, any authentication or secrecy property—there exists a counterexample that involves no secondary strands. Hence, the secondary strands cannot cause any authentication or secrecy property to fail, if that property would hold of the primary protocol in isolation.

Proposition 5.10 (Protocol Independence) *Suppose that Σ is closed, full, and has disjoint encryption. If $\mathcal{C}$ is any bundle in Σ, then there is a bundle $\mathcal{C}'$ in Σ that is equivalent to $\mathcal{C}$ such that $\mathcal{C}' \cap \Sigma_2 = \emptyset$.*

PROOF. By the Penetrator Normal Form Lemma 4.6, we may assume that $\mathcal{C}$ is normal. In addition, by Proposition 4.12, we may assume that each bridge in $\mathcal{C}$ is simple, and by Proposition 4.14, we may assume that each bridge in $\mathcal{C}$ is minimal.

By Proposition 5.9, if p is any in-bound linking path, then $\mathrm{pbt}(p)$ does not originate uniquely on a primary strand. Hence, by Proposition 3.5, $\mathrm{pbt}(p)$ is public in Σ.

Not let $n_j \to n_{j+1}$ be any bridge in $\mathcal{C}$ such that there exists an in-bound linking path p traversing $n_j \to n_{j+1}$. Applying Proposition 3.6 yields a bundle $\mathcal{C}_1$ containing a node m'_p, such that:

1. $\mathrm{term}(m'_p) = \mathrm{term}(n_{j+1})$, and
2. for all nodes m_0, if $m_0 \preceq m'_p$ then m_0 is a penetrator node.

We may apply Proposition 3.10 to this node m'_p, n_{j+1}, and n_j. In the resulting bundle $\mathcal{C}_2$, $m_0 \prec n_{j+1}$ only if $m_0 \preceq m'_p$. Hence, no linking path in $\mathcal{C}_2$ traverses n_{j+1}, and every linking path in $\mathcal{C}_2$ was already a linking path in $\mathcal{C}$.

Applying this construction to every bridge $n_j \to n_{j+1}$ in $\mathcal{C}$ such that there exists an in-bound linking path p traversing $n_j \to n_{j+1}$, we obtain a bundle $\mathcal{C}_n$ in which there are no in-bound linking paths.

If we let $\mathcal{C}' = \mathcal{C}_n \mid \Sigma_1$, then by Proposition 3.7 $\mathcal{C}'$ is a bundle; clearly $\mathcal{C}'$ is equivalent to $\mathcal{C}_i$. Moreover $\mathcal{C}' \cap \Sigma_2 = \emptyset$, because there is no $n \in \Sigma_2$ and $m \in \Sigma_1$ such that $n \preceq_{\mathcal{C}'} m$, as the latter would entail that there exists a linking path. ■

References

[1] Martín Abadi and Roger Needham. Prudent engineering practice for cryptographic protocols. In *Proceedings, 1994 IEEE Symposium on Research in Security and Privacy*, pages 122–136. IEEE, IEEE Computer Society Press, 1994.

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

[6] Shimon Even, Oded Goldreich, and Adi Shamir. On the security of ping-pong protocols when implemented using the RSA. In *Advances in Cryptology—*CRYPTO *'85*, LNCS, pages 58–72. Springer Verlag, 1985.

[7] Li Gong and Paul Syverson. Fail-stop protocols: An approach to designing secure protocols. In *5th International Working Conference on Dependable Computing for Critical Applications*, pages 44–55, September 1995.

[8] John Kelsey, Bruce Schneier, and David Wagner. Protocol interactions and the chosen protocol attack. In *Security Protocols, International Workshop April 1997 Proceedings*, pages 91–104. Springer-Verlag, 1998.

[9] 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.

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

[11] 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.

[12] Catherine Meadows. Analysis of the Internet Key Exchange protocol using the NRL protocol analyzer. In *Proceedings, 1999 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 1999.

[13] 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.

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

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

[16] F. Javier Thayer Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 1999. Forthcoming.

[17] D. Wagner and B. Schneier. Analysis of the ssl 3.0 protocol. In *Proceedings, Second USENIX Workshop on Electronic Commerce*, pages 29–40, 1996. Available at http://www.counterpane.com/ssl.html.

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