Protocol Independence through Disjoint Encryption*

Joshua D. GUTTMAN F. Javier THAYER Fábrega

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

Abstract

One protocol (called the primary protocol) is independent *of other protocols (jointly called the secondary protocol) if the question whether the primary protocol achieves a security goal never depends on whether the secondary protocol is in use.*

In this paper, we use multiprotocol strand spaces ([27], cf. [28]) to prove that two cryptographic protocols are independent if they use encryption in non-overlapping ways. This theorem (Proposition 7.2) applies even if the protocols share public key certificates and secret key "tickets."

We use the method of [8, 7] to study penetrator paths*, namely sequences of penetrator actions connecting regular nodes (message transmissions or receptions) in the two protocols. Of special interest are* inbound linking paths*, which lead from a message transmission in the secondary protocol to a message reception in the primary protocol. We show that bundles can be modified to remove all inbound linking paths, if encryption does not overlap in the two protocols. The resulting bundle does not depend on any activity of the secondary protocol. We illustrate this method using the Neuman-Stubblebine protocol as an example [21, 27].*

1 Introduction

Whether a cryptographic protocol achieves a security goal depends on what cannot happen. To authenticate a regular principal engaging in a protocol run, we must observe a pattern of messages that can only be constructed by that principal in that run, regardless of how the penetrator combines his own actions with those of principals engaging in other runs [5]. When several cryptographic protocols are combined, the penetrator has new opportunities to obtain the messages which ought to authenticate principals to their peers. Indeed, because protocol mixing has shown itself to be a significant cause of protocol failure, and makes protocol analysis more difficult [2, 6, 12, 19, 27, 29], it has been identified [20] as a key problem in applying formal methods to cryptographic protocols.

Moreover, in practice, different protocols using cryptography are usually combined. A key distribution protocol is useful only if the session key it delivers is used for encryption. That later use may involve constructing messages similar to messages used in the key distribution protocol itself. Does this make replay attacks possible? Does the use of a key undermine the guarantees provided by the protocol distributing that key?

There are other reasons why protocol mixture is prevalent. Many recent protocols have large numbers of different options, and therefore have large numbers of different subprotocols [18, 9, 4, 19]. Each of these protocols may be easy to analyze on its own. But the same principal is required to be able to engage in any sub-protocol. Can the penetrator manipulate this willingness for his own purposes?

When protocols are mixed together, and we want to appraise whether the security of one is affected by the others, we will refer to the protocol under study as the *primary* protocol. We will refer to the others as *secondary* protocols.

Common sense suggests a rule of thumb when protocols are to be mixed together. This rule is that if the primary protocol uses a particular form of encrypted message as a test to authenticate a peer [7], then the secondary protocols should not construct a message of that form. The sets of encrypted messages that the different protocols handle should be disjoint. One way to arrange for this is to give each protocol some distinguishing value, such as a number; that number may then be included as part of each plaintext before encipherment. Then no principal can mistake a value as belonging to the wrong protocol. Another way to achieve disjoint encryption is to ensure that different protocols never use the same key, although this may be expensive or difficult to arrange.

Although the Abadi-Needham paper on prudent engineering practice for cryptographic protocols [1] does not

*This work was supported by the National Security Agency through US Army CECOM contract DAAB07-99-C-C201. Appears in *Proceedings, 13th IEEE Computer Security Foundations Workshop*, Cambridge, July 2000.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
10 APR 2000		**00-04-2000 to 00-04-2000**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Protocol Independence through Disjoint Encryption	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **MITRE Corporation,202 Burlington Road,Bedford,MA,01730-1420**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**12**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

discuss mixing different protocols, this rule—to try to achieve disjoint encryption—is in the same spirit as those it proposes.

In this paper we will prove that, properly formalized, it suffices. If two protocols have disjoint encryption, then the first protocol is independent of the second. By this we mean that if the first protocol achieves a security goal (whether an authentication goal or a secrecy goal [28]) when the protocol is executed in isolation, then it still achieves the same security goal when executed in combination with the second protocol. One of the advantages of our approach is that the result works for all secrecy and authentication goals; in this it continues a trend visible from several recent papers [16, 11, 26, 25, 10].

Section 2 introduces some background, summarizing the basic ideas and notation of strand spaces (with more detail in Appendix A and [28]). Section 3 introduces some notions not used in [28]; *multiprotocol strand spaces* were introduced in [27], and *new components* are emphasized in [7].

Section 4 studies paths through bundles, and introduces two special forms for bundles, in which the penetrator avoids roundabout activities; additional detail and proofs may be found in [8]. In the remainder of the paper we study bundles of these special forms in multiprotocol strand spaces, focusing on the relation between events in the primary protocol and events in the secondary protocol. Section 5 considers the private values that the primary protocol assumes will not be guessed. Section 6 defines our technical notion of disjoint encryption and Section 7 proves the protocol independence theorem, of which we give an application in Section 8.

2 Strand Spaces

Terms 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.1.)

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.2.)

Strand Spaces, Origination, and Bundles 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, $n \Rightarrow^+ n'$ [respectively, $n \Rightarrow^* n'$] means that $n = \langle s, i\rangle$ and $n' = \langle s, j\rangle$ for some $j > i$ [respectively, for some $j \geq i$]. The relation $n \rightarrow n'$ represents inter-strand communication; it means that $\mathrm{term}(n_1) = +t$ and node $\mathrm{term}(n_2) = -t$.

A *strand space* Σ is a set of strands. 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. 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.

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. 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. This relation expresses the fact that n_1 causally contributes to n_2 occurring in $\mathcal{C}$. (See Definitions A.5, A.7.)

Regular Strands and Penetrator Strands 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 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.)

3 New Components and Multiprotocol Strand Spaces

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* (Definition A.2). Components are either atomic values or encryptions. 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. For instance, the three components of the concatenated term

$$B \, \{|N_a \, K \, \{|K \, N_b|\}_{K_B}|\}_{K_A} \, N_a$$

are B, $\{|N_a \, K \, \{|K \, N_b|\}_{K_B}|\}_{K_A}$, and N_a.

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.2). A component is new even if it has occurred earlier as a nested subterm of some larger component $\cdots \{|\cdots t \cdots|\}_K \cdots$. We say t is a component of n if t is a component of term(n). 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 [24].

To represent multiple protocols [27], we select some regular strands as being runs of the primary protocol; we call these strands *primary strands*.

Definition 3.1 *A* multiprotocol strand space *is a strand space (Σ, tr) together with a distinguished subset of the regular strands $\Sigma_1 \subset \Sigma \setminus \mathcal{P}_\Sigma$ called the set of* primary *strands.*

Σ_2 denotes the set of all other regular strands, called *secondary strands*. A node is primary or secondary if the strand it lies on is. From the point of view of a particular analysis, the secondary strands represent runs of other protocols, different from the primary one under analysis.

Two bundles are *equivalent* if they have the same primary nodes.

Definition 3.2 *Two bundles $\mathcal{C}, \mathcal{C}'$ in the multiprotocol strand space (Σ, tr, Σ_1) are equivalent if and only if, for every node $n \in \Sigma_1$, $n \in \mathcal{C}$ iff $n \in \mathcal{C}'$.*

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

Agreement and non-injective agreement properties [15, 28, 30] 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 nodes of a protocol strand (for instance, a responder strand), then it also contains matching nodes of another strand (for instance, an initiator strand using the same data values). As such, it always concerns what primary nodes must be present in bundles. Penetrator activity or secondary nodes 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 primary strands. When stated in this form, the assertion that a value is uncompromised is invariant under bundle equivalences.

4 Paths, Normal Bundles, Efficient Bundles

We will now introduce the *paths* through bundles, and examine some special forms of bundle, such that every bundle is equivalent (in our sense) to a bundle in each of these special forms. [8] contains the proofs that we omit here. The notation $m \longmapsto n$ means:

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

A *path* p through $\mathcal{C}$ is any finite sequence of nodes and edges $n_1 \longmapsto n_2 \longmapsto \cdots \longmapsto n_k$. We refer to the ith node of the path p as p_i. The length of p is $|p|$, and we write $\ell(p)$ to mean $p_{|p|}$, i.e. the last node in p.

Clearly, $p_1 \preceq_{\mathcal{C}} \ell(p)$ whenever there is such a path p with zero or more arrows. The converse is not true. For instance, if m and n lie on the same strand with $m \Rightarrow^+ n$ and m is positive or n is negative, then we do not have $m \longmapsto n$. Unless there happens to be some other path from m to n, we have $m \preceq_{\mathcal{C}} n$ without any path from m to n.

Proposition 4.1 *Let $\mathcal{C}$ be a bundle and $m \preceq_{\mathcal{C}} n$. Then there is a path p where $m \Rightarrow^* p_1$ and $\ell(p) \Rightarrow^* n$.*

PROOF. If m and n lie on the same strand, then there is the path p with $|p| = 1$ and $p_1 = m$. So assume (inductively) that the proposition holds for all $n' \prec n$. Because $m \preceq n$, there is a sequence of arrows $\rightarrow$ and $\Rightarrow$ from m to n. If the last arrow is $n' \Rightarrow n$, then (inductively) there is a path p' with $m \Rightarrow^* p'_1$ and $\ell(p) \Rightarrow^* n'$, so that $\ell(p) \Rightarrow^+ n$.

Suppose that the last arrow is $n' \rightarrow n$. If $\ell(p')$ is negative, we may adjoin the two arrows $\ell(p') \Rightarrow^+ n' \rightarrow n$. Suppose next that $\ell(p')$ is positive. If $\ell(p') = n'$, we may adjoin the arrow $n' \rightarrow n$ to obtain the desired p. If $m \Rightarrow^* \ell(p')$, then we may take $p = n' \rightarrow n$. Otherwise, p' is of the form $\cdots \rightarrow n'' \Rightarrow^+ \ell(p')$, so we may take $p = \cdots \rightarrow n'' \Rightarrow^+ n' \rightarrow n$. ■

Proposition 4.2 *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. *$m_1 \Rightarrow m_2$ if $m_1, m_2 \in G$ and $m_1 \Rightarrow m_2$ in $\mathcal{C}$.*

Then G is a bundle, called $\mathcal{C} \mid N$. Moreover, if $\mathcal{C} \cap \Sigma_1 \subset N$, $\mathcal{C} \mid N$ is equivalent to $\mathcal{C}$.

PROOF. Clearly G is finite and acyclic, as it is a subgraph of $\mathcal{C}$. Suppose that $n_2 \in G$; we want to show that Clause 1 in the definition of bundle holds. Because $\mathcal{C}$ is a bundle, there is a unique n_1 in $\mathcal{C}$ such that $n_1 \rightarrow n_2$ in $\mathcal{C}$; by Clause 1 $n_1 \in G$; by Clause 2, $n_1 \rightarrow n_2$ in G.

Suppose next that $n_2 \in G$ and $n_1 \Rightarrow n_2$. Then $n_1 \in G$ by Clause 1, because $n_1 \preceq_{\mathcal{C}} n_2$. By Clause 3, the $\Rightarrow$ relation holds between them in G also. So Clause 2 in the definition of bundle holds.

Suppose that $\mathcal{C} \cap \Sigma_1 \subset N$. Since we always have $n \preceq_{\mathcal{C}} n, \mathcal{C} \cap \Sigma_1 \subset G$. Since every node in G is in $\mathcal{C}$, we may infer that $G \cap \Sigma_1 \subset \mathcal{C} \cap \Sigma_1$, so that $\mathcal{C}$ and $\mathcal{C} \mid N$ have the same primary nodes. ■

From Propositions 4.1 and 4.2 it follows that if $\mathcal{C}$ has no path leading from a secondary node to a primary node, then the secondary nodes are irrelevant, because $\mathcal{C} \mid \Sigma_1$ is equivalent to $\mathcal{C}$ but has no secondary nodes. We call such a path an *inbound linking path*. Conversely, if $p_1 \in \Sigma_1$ and $\ell(p) \in \Sigma_2$, then p is an *outbound linking path*. We have thus taken the point of view of the primary protocol, because the results of this paper are not symmetrical between the two protocols.

Unless otherwise indicated, we henceforth assume all paths begin on a positive node, and end on a negative node. Given a path p, one $\Rightarrow^+$ edge immediately precedes another $\Rightarrow^+$ edge in p if they are separated in p by a single $\rightarrow$ edge.

Definition 4.3 *A path p is a* penetrator path *if p_i is a penetrator node whenever $i \neq 1$ or $|p|$.*

A $\Rightarrow^+$-edge on a penetrator strand is constructive *if it lies on an* E *or* C *strand. It is* destructive *if it lies on a* D *or* S *strand.*

Any other penetrator node lies on a K or M node, and is called an *initial* node. By analogy with Prawitz's notion of normal derivation [23], we define:

Definition 4.4 *A bundle $\mathcal{C}$ is* normal *if, for any penetrator path of $\mathcal{C}$, every destructive edge precedes every constructive edge.*

In [8] we show a result akin to one in [3]:

Proposition 4.5 (Penetrator Normal Form Lemma) *For any bundle $\mathcal{C}$ there exists an equivalent normal bundle $\mathcal{C}'$.*

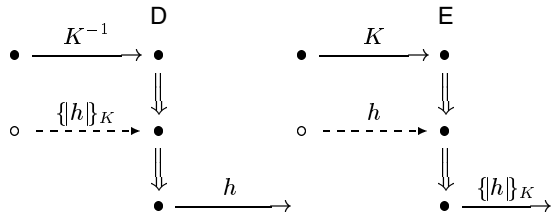


Figure 1. Entering a D or E strand through a key edge

4.1 Rising and Falling Paths

Normal bundles are more predictable than bundles in general because the penetrator never builds up values just to take them apart again. In particular, certain penetrator paths in a normal bundle have a natural relation to the structure of the terms that they manipulate.

Definition 4.6 *A penetrator path is* falling *if for all adjacent nodes $n \longmapsto n'$ on the path $term(n') \sqsubset term(n)$. It is* rising *if for all adjacent nodes $n \longmapsto n'$ on the path $term(n) \sqsubset term(n')$.*

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 1). Call the edge labeled K^{-1} in Figure 1 a D-key edge. The other incoming edge into a D strand is a D-cyphertext edge.

Paths entering an encryption strand through the key transmission edge (Figure 1) are symmetrical. We refer to a E-key edge and an E-plaintext edge. In this case we have a stronger conclusion, because a constructive p can traverse an E-key edge only once, along the edge $p_1 \rightarrow p_2$, and only if $\text{term}(p_1) \in \mathsf{K}$. After that we have a compound term, not an atomic key.

Proposition 4.7 *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 $term(p_1) \notin \mathsf{K}$.

By examining the destructive strands, and using induction, we may infer:

Proposition 4.8 *Suppose that p is a falling penetrator path, and $term(p_i) = t$ where t is simple. Then for some j with $1 \leq j \leq i$, $t \sqsubset term(p_j)$ and $term(p_j)$ is a component of p_1.*

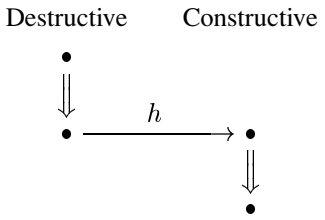


Figure 2. Internal Bridge

4.2 Transformation Paths

Definition 4.9 *A* transformation path *is a path for which each node p_i is labelled by a component $\mathcal{L}_i$ of p_i in such a way that $\mathcal{L}_i = \mathcal{L}_{i+1}$ unless $p_i \Rightarrow^+ p_{i+1}$ and $\mathcal{L}_{i+1}$ is new on the strand of p_{i+1}.*

In the following result [8, Proposition 3.18], the path p may terminate on a positive node.

Proposition 4.10 *Suppose that Σ is a strand space, and $\mathcal{C}$ a bundle in Σ. If $m \in \mathcal{C}$, $a \sqsubset t$ and t is a component of m, then there exists a transformation path $(p, \mathcal{L})$ through $\mathcal{C}$ such that*

1. *a originates on p_1, while $\ell(p) = m$ and $\mathcal{L}_{|p|} = t$,*
2. *$a \sqsubset \mathcal{L}_j$ for all $j = 1$ to $|p|$, and*
3. *p never traverses the key node of an E-strand or D-strand.*

Moreover, $\mathcal{L}_{j-1} \sqsubset \mathcal{L}_j = term(p_j)$ if p_j is a positive E-node, and $\mathcal{L}_j \sqsubset \mathcal{L}_{j-1} = term(p_{j-1})$ if p_j is a positive D-node, while $\mathcal{L}_j = \mathcal{L}_{j-1}$ if p_j is a positive C-node or S-node.

4.3 Bridges

All destructive edges precede constructive edges in a normal penetrator path. The edge that separates the destructive portion of a path from the constructive portion is of special interest. We call it a bridge.

Definition 4.11 *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 3–2.*

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 $g\,h$.*

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.

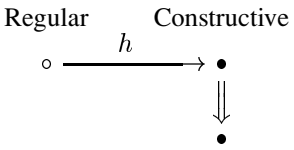


Figure 3. Entry Bridge

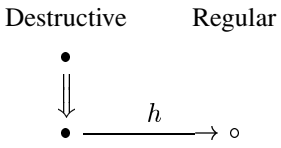


Figure 4. Exit Bridge

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

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

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, much of Section 7 is simpler with the assumption of simple bridges.

Proposition 4.13 *Suppose $\mathcal{C}$ be a normal bundle with simple bridges. If $(p, \mathcal{L})$ is a transformation path in $\mathcal{C}$ where p is a penetrator path which starts at a bridge, then there is smallest index α such that term$(p_\alpha) = \mathcal{L}_i = \mathcal{L}_{|p|}$ whenever $\alpha \leq i \leq |p|$. Moreover, if $\mathcal{L}$ is not constant then p_α is the positive node of an E-strand.*

Thus if p starts at a bridge, there is always an index α such that term$(p_\alpha) = \mathcal{L}_{|p|}$.

Similarly, if $(p, \mathcal{L})$ is a transformation path in $\mathcal{C}$ where p is a penetrator path which ends at a bridge, then either $\mathcal{L}$ is

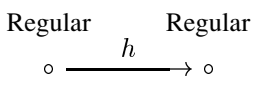


Figure 5. External Bridge

constant or there is a smallest index β such that $\mathcal{L}_\beta \neq \mathcal{L}_1$. p_β is the positive node of a D*-strand and term$(p_{\beta-1}) = \mathcal{L}_{\beta-1}$.*

Thus if p ends at a bridge, there is always an index β such that 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 term(p_{i+1}) is an encrypted term and therefore term(p_{i+1}) has only one component. Therefore, 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 term$(p_i) = \mathcal{L}_i$.

Notice that if $\mathcal{L}$ is constant and p_i is a bridge node, the simple bridges assumption implies term(p_i) consists of a single component. Clearly, $\mathcal{L}_1 = \mathcal{L}_{|p|} = \mathcal{L}_i = \text{term}(p_i)$. ■

4.4 Efficient Bundles

Definition 4.14 *A bundle is* efficient *if and only if, for every node m and negative 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 can get from the node m, rather than making use of additional, unnecessary regular nodes such as m'. In [8] we prove:

Proposition 4.15 *Any bundle $\mathcal{C}$ is equivalent to an efficient bundle $\mathcal{C}'$. Moreover, $\mathcal{C}'$ may be chosen to be normal and to have simple bridges.*

Proposition 4.16 *Suppose $\mathcal{C}$ is a normal efficient bundle with simple bridges and $(p, \mathcal{L})$ $(p', \mathcal{L}')$ are transformation paths in $\mathcal{C}$. Assume p is a penetrator path which starts at a bridge, p' is a penetrator path which ends at a bridge 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 4.13, there are indices α, β such that term$(p_\alpha) = \mathcal{L}_{|p|}$ and term$(p'_\beta) = \mathcal{L}'_1$. In particular, $p_\alpha \prec m \prec p'_\beta$ and term(p_α), term(p'_β) both have single components. Therefore, by bundle efficiency, term$(p_\alpha) \neq$ term(p'_β). In particular, $\mathcal{L}'_1 \neq \mathcal{L}_{|p|}$. ■

5 Public Values and Full Spaces

For what values does privacy matter? Which values should the penetrator be assumed not to know initially, and not to be lucky enough to guess?

By a *security goal*, we mean a theorem about authentication or secrecy [28, Section 8.2]. A security goal is typically a universally quantified implication, concerning every strand space Σ of a particular kind, every bundle $\mathcal{C}$ in Σ, and every choice of additional parameters that determine particular principals, keys, and data values. The implication takes the form:

if $\mathcal{C}$ contains primary nodes matching certain templates, and some conditions hold on the parameters,

then some additional nodes must exist in $\mathcal{C}$ (in the case of an authentication goal), or must not exist in $\mathcal{C}$ (in the case of an secrecy goal).

The conditions on the parameters frequently stipulate that a value should be unknown to the penetrator, or that it should be chosen unpredictably. When a value is subject to an assumption of this kind, let us call that value a *privacy value* for the security goal. We will also call a set of nodes that instantiate the primary node templates in the antecedent for a choice of values for the parameters a *core node set* for the security goal. The security goal is "talking about" strand spaces and bundles including the core node set, in which the conditions on the parameters hold true.

Examination of a variety of security goals [28, 27, 7] for different protocols suggests that there are two types of assumptions about privacy values:

1. Assumptions about long term keys, which are used for encryption in a protocol, but never uttered as a subterm of any message;
2. Assumptions about values originating uniquely on some primary strand of the protocol.

We will call the values involved *long term privacy values* and *fresh privacy values* respectively.

Suppose that we are considering a particular security goal, and have selected a core node set. Many different strand spaces will contain these nodes, for instance if they differ only in their penetrator strands, especially penetrator M-strands and K-strands, which are the strands that determine whether a privacy value is given to the penetrator. So long as we do not add strands that falsify a privacy assumption for some parameter used in the given core node set, we may freely add M-strands and K-strands to a space Σ. Adding strands that do not falsify privacy assumptions cannot convert a space Σ, to which some security goal applies, into a space Σ' in which the assumptions of that goal are not

met. If a privacy assumption is already false in Σ', then no penetrator strands we add can make a difference to it.

Moreover, a privacy assumption *is* false whenever a privacy value originates on a secondary strand of Σ. Hence, adding an M-strand or K-strand for this value cannot falsify any privacy assumption that is satisfied in Σ. We regard a space Σ as *full* when all of these harmless M-strands and K-strands are present in Σ:

Definition 5.1 *A strand space Σ is* full *if every atomic value $a \in \mathsf{T} \cup \mathsf{K}$ that originates on any secondary strand in Σ also originates on some M-strand or K-strand in Σ.*

An atomic value $a \in \mathsf{T} \cup \mathsf{K}$ is private in Σ if a never originates on a secondary or penetrator strand in Σ. Otherwise, it is public. A concatenated value $g\,h$ is public if g and h are. An encrypted value $\{|h|\}_K$ is public if h and K are.

Observe that if Σ is full, then t is public if and only if there is a bundle $\mathcal{C}$ consisting only of penetrator strands and containing a node with term t.

Definition 5.2 *A bundle $\mathcal{C}$ is* standard *if*

1. *$\mathcal{C}$ is normal, efficient, and has simple bridges; and*
2. *If an atomic value $a \in \mathsf{T} \cup \mathsf{K}$ originates on any secondary node in $\mathcal{C}$, then a also originates on some penetrator node $n_a \in \mathcal{C}$; if term$(m) = -a$, then $n_a \to m$.*

Clause 2 is a way of stating that if some principal executing a secondary protocol is lucky enough to guess the value a, then the penetrator may be that lucky, too, and we may suppose that the penetrator supplies it to any consumer.

Proposition 5.3 *If Σ is a full strand space and $\mathcal{C}$ is a bundle in Σ, then there is a standard bundle $\mathcal{C}'$ in Σ such that $\mathcal{C}'$ is equivalent to $\mathcal{C}$.*

PROOF. Clause 1 holds by Propositions 4.5, 4.12, and 4.15.

To establish Clause 2, observe that there are finitely many secondary nodes in $\mathcal{C}$. Only finitely many values $a \in \mathsf{T} \cup \mathsf{K}$ may originate at each, so the values originating on secondary nodes form a finite set S. Thus, we may add finitely many M and K strands to $\mathcal{C}$, originating each $a \in S$; these strands must exist in Σ because Σ is full. We refer to the new penetrator node originating $a \in S$ as n_a. If $n \in \mathcal{C}$ is a negative node with term$(n) = a \in S$, then there is a unique m such that $m \to n$. We replace this arrow with $n_a \to n$. Hence Clause 2 is satisfied in the resulting bundle. ■

6 Disjoint Encryption

The simplest way to state the disjoint encryption assumption would be to require that the two protocols not use the same ciphertext as a part of any message. That would mean that if $n_1 \in \Sigma_1$ and $n_2 \in \Sigma_2$, and if $\{|h|\}_K \sqsubset \text{term}(n_1)$, then $\{|h|\}_K \not\sqsubset \text{term}(n_2)$.

However, this simple version is unnecessarily restrictive. The secondary protocol would be unable to accept public-key certificates generated in the primary protocol, which is intuitively harmless because the contents are public in any case. The secondary protocol would also be unable to reuse symmetric-key tickets such as those generated by the Kerberos Key Distribution Center [13]. These are also intuitively harmless, so long as the secondary protocol does not extract private values from within them, or repackage their private contents, potentially insecurely. Hence we allow these harmless exceptions to the requirement that no encrypted term be used by both protocols.

Definition 6.1 (Disjoint Outbound Encryption) Σ *has* disjoint outbound encryption *if and only if the following always holds. Suppose given a positive node $n_1 \in \Sigma_1$ and a negative $n_2 \in \Sigma_2$, and private $a \sqsubset \{|h|\}_K$ such that $\{|h|\}_K \sqsubset$ term(n_1) and $\{|h|\}_K \sqsubset$ term(n_2).*

Then there is no positive n_2' such that $n_2 \Rightarrow^+ n_2'$ and a occurs in a new component of n_2'.

This definition has the important property that atomic values cannot "zigzag" back and forth from primary to secondary nodes, before being disclosed to the penetrator.

Proposition 6.2 (No Zigzags) *Let Σ have disjoint outbound encryption, and let $\mathcal{C}$ be a standard bundle in Σ. Suppose $(p, \mathcal{L})$ is a transformation path such that term$(\ell(p)) = -a$ where $a \in \mathsf{K} \cup \mathsf{T}$, $a \sqsubset \mathcal{L}_i$ for all $1 \leq i \leq |p|$, and $p_k \in \Sigma_2$. Then $p_j \notin \Sigma_1$ for $j < k$.*

In particular, a is not private.

PROOF. Argue by contradiction and suppose that $p_j \in \Sigma_1$ with $j < k$. If we choose j to be the greatest such value, and assume k chosen to be the least number $> j$ such that $p_k \in \Sigma_2$, then $p_j \longmapsto \cdots \longmapsto p_k$ is a penetrator path. Since $\mathcal{C}$ is standard, $p_j \longmapsto \cdots \longmapsto p_k$ has a simple bridge $p_\beta \longrightarrow p_{\beta+1}$, so $a \sqsubset \text{term}(p_\beta) = \mathcal{L}_\beta$. Since $a = \text{term}(\ell(p))$, by efficiency, $a \neq \text{term}(p_\beta)$. Thus, $a \sqsubset e = \text{term}(p_\beta)$, where e is an encrypted unit $\{|h|\}_K$.

Let $\gamma > k$ be the smallest index such that $\mathcal{L}_\gamma \neq \mathcal{L}_{\gamma+1}$. The node p_γ cannot be a penetrator node, because then $\mathcal{L}_\gamma = \text{term}(p_\gamma) = p_{k-1}$, which contradicts Proposition 4.16. If $\mathcal{L}_\gamma \in \Sigma_1$, then there is a penetrator path leading to it, again violating Proposition 4.16. Therefore, $p_\gamma \in \Sigma_2$.

Since $p_{\gamma+1}$ has a new component $\mathcal{L}_{\gamma+1}$ with subterm a, by outbound disjoint encryption, a is not private. By the

definition of standard bundle (Definition 5.2, Clause 2), $\mathcal{C}$ contains an M- or K-node n_a such that $a = \text{term}(n_a)$, and $n_a \to \ell(p)$, contradicting our choice of p.

To infer that a is not private, use Proposition 4.10 to obtain a transformation path $(p, \mathcal{L})$ such that a originates at p_1, and apply the preceding. ■

For inbound linking paths, we must also choose the exceptions to naïve disjoint encryption. We stipulate that encrypted units $\{|h|\}_K$ may be exceptions if they are not included in a new components of a secondary node, but are emitted only in the same form in which they were received by the secondary protocol previously. When a positive secondary node emits an exception, the component must have been received previously on the same strand, and not newly manufactured and potentially useful to the penetrator.

Definition 6.3 (Disjoint Encryption) Σ *has* disjoint inbound encryption *if, for all negative* $n_1 \in \Sigma_1$ *and positive* $n_2 \in \Sigma_2$*, and for all* $\{|h|\}_K$*, if* $\{|h|\}_K \sqsubset \mathit{term}(n_1)$ *and* $\{|h|\}_K \sqsubset \mathit{term}(n_2)$*, then* $\{|h|\}_K \not\sqsubset t_0$*, for any new component* t_0 *of* n_2.

Σ *has* disjoint encryption *if it has both disjoint inbound encryption and disjoint outbound encryption.*

7 The Protocol Independence Theorem

Definition 7.1 Σ_1 *is independent of* Σ_2 *if for every bundle in* Σ*, there is a bundle* $\mathcal{C}'$ *in* Σ *that is equivalent to* $\mathcal{C}$ *such that* $\mathcal{C}'$ *is disjoint from* Σ_2.

Proposition 7.2 (Protocol Independence) *If* Σ *is full, and has disjoint encryption, then* Σ_1 *is independent of* Σ_2.

PROOF. By Proposition 5.3, we may assume that $\mathcal{C}$ is standard. We want to show that there are no inbound linking paths in $\mathcal{C}$.

Let p be an inbound linking path. Suppose first that p traverses an atomic value $a \in \mathsf{T} \cup \mathsf{K}$. This may either be the key edge into a D or E strand, or it may be the bridge of p. In any case, let a be the first atomic value on p. If a is public, then because $\mathcal{C}$ is standard (Definition 5.2), Clause 2 contradicts the assumption that p is an inbound linking path. Therefore a would have to be private, but that contradicts Proposition 6.2.

Suppose next that p never traverses an atomic value. Then in particular it never traverses a key edge into a D or E strand. Thus, the path bridge term $\text{pbt}(p) \sqsubset \text{term}(p_1)$ and $\text{pbt}(p) \sqsubset \text{term}(\ell(p))$. Since $\text{pbt}(p)$ is not atomic but it is simple, it is of the form $\{|h|\}_K$. Therefore, by disjoint inbound encryption, it does not occur in a new component of p_1. If $a \sqsubset t$ where t is a component of p_1, then there is $m \Rightarrow^+ p_1$ such that t is a component of m. Since (Proposition 4.8) there is a node p_i such that $\text{term}(p_i) = t$, the relations $m \prec p_1 \prec p_i$ contradict efficiency.

Therefore there is no inbound linking path p in any standard bundle $\mathcal{C}$. It follows that there are no $n_1 \in \mathcal{C} \cap \Sigma_1$ and $n_2 \in \mathcal{C} \cap \Sigma_2$ such that $n_2 \preceq_{\mathcal{C}} n_1$, because by Proposition 4.1, there would be a path with $n_2 \Rightarrow^* p_1$ and $\ell(p) \Rightarrow^* n_1$, and p would be an inbound linking path. Hence, we may apply Proposition 4.2, letting $N = \Sigma_1$. ■

An easy consequence of this theorem show that if the primary and secondary protocols share no keys whatever, then we have independence.

Corollary 7.3 *Let* Σ *be full. For* $i = 1$ *and* 2*, let* $\mathfrak{K}_i$ *be the set of* K *such that* $K \sqsubset \mathit{term}(n)$ *for any* $n \in \Sigma_i$ *or* $\{|h|\}_K \sqsubset \mathit{term}(n)$ *for any* h *and any* $n \in \Sigma_i$.

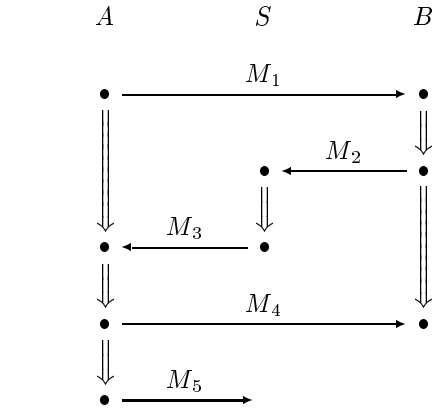
If $\mathfrak{K}_1 \cap \mathfrak{K}_2 = \emptyset$*, then* Σ_1 *is independent of* Σ_2.

In realistic situations, if Σ_1 and Σ_2 involve the activity of different principals, and the keys for the protocols are chosen in an unpredictable way from a large set, then the keys they use will never overlap. Therefore, Σ_1 is independent of Σ_2. The same holds when the same principals may participate in both protocols, but they choose keys independently for each protocol.

8 An Application of Protocol Independence

The familiar Neuman-Stubblebine protocol [21] will illustrate the usefulness of the Protocol Independence Theorem. It 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 6. This session key is embedded in a re-usable ticket of the form $\{|A\, K\, T|\}_{K_{B\,S}}$. In the re-authentication sub-protocol, the key distribution center no longer needs to be involved; the initiator presents the same ticket again to the responder, as shown in Figure 7 on the left. We have added a fictitious message $B\, \{|A\, K\, T|\}_{K_{B\,S}}$, which is sent by a strand of the authentication protocol and received by a strand of the re-authentication protocol. It represents a portion of the client's state in the implementation. Clearly, representing this internal state as a visible message could only add vulnerabilities not conceal them.

We regard the re-authentication protocol as the secondary protocol; the presence of the re-authentication protocol should not undermine any security guarantee offered by the primary protocol. However, terms of the form $\{|N|\}_K$ are constructed as new components on secondary strands, and accepted on primary strands. Hence the corresponding multiprotocol strand space does not have disjoint inbound encryption. Indeed, the penetrator can use a session of the re-authentication protocol to complete a responder strand in a bundle with no initiator [27].



$M_1 = A\, N_a$

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

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

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

$M_5 = B\, \{|A\, K\, t_b|\}_{K_{BS}}$

Figure 6. Neuman-Stubblebine Part I (Authentication)

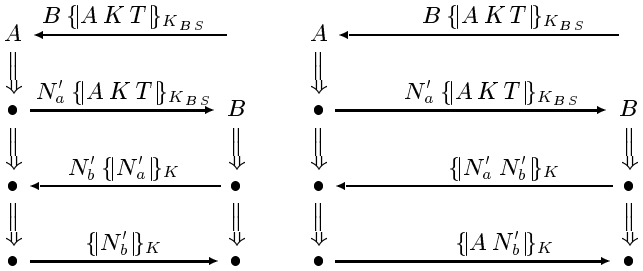


Figure 7. Neuman-Stubblebine Part II, original and modified

For this reason, we amend the re-authentication protocol to the form shown on the right of Figure 7 [27]. To apply our independence theorem, we check that the corresponding strand space Σ has disjoint encryption. But that is trivial, because tickets $\{|A\, K\, T|\}_{K_{BS}}$ are the only common encrypted subterms of primary and secondary nodes. The outbound property holds because no private subterm of a ticket is uttered in a new component of a secondary node. The inbound property holds because no new component of a secondary node contains a ticket.

Therefore, if Σ is a full strand space and $\mathcal{C}$ is a counterexample to some security property, we may deform $\mathcal{C}$ into an equivalent standard bundle $\mathcal{C}'$, in which there are no secondary nodes. $\mathcal{C}'$ is still a counterexample, assuming that the security property is invariant under bundle equivalences, as authentication and secrecy properties are. Thus, if the primary protocol fails to meet the security goal, that is independent of the presence of the secondary protocol: the corrected Neuman-Stubblebine re-authentication protocol is entirely guiltless in this affair.

Acknowledgments We are grateful to Sylvan Pinsky and Al Maneki for encouragement, support, and many technical discussions.

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

[3] 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.

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

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

[6] 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.

[7] 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.

[8] Joshua D. Guttman and F. Javier THAYER Fábrega. Authentication tests and the normal penetrator. MTR 00B04, The MITRE Corporation, February 2000. Also submitted for publication.

[9] D. Harkins and D. Carrel. *The Internet Key Exchange (IKE)*. IETF Network Working Group RFC 2409, November 1998.

[10] James Heather, Gavin Lowe, and Steve Schneider. How to prevent type flaw attacks on security protocols. In *Proceedings, 13th Computer Security Foundations Workshop*. IEEE Computer Society Press, July 2000.

[11] Mei Lin Hui and Gavin Lowe. Safe simplifying transformations for security protocols. In *12th Computer Security Foundations Workshop Proceedings*, pages 32–43. IEEE Computer Society Press, June 1999.

[12] 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.

[13] J. Kohl and C. Neuman. The Kerberos network authentication service (v5). RFC 1510, September 1993.

[14] 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.

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

[16] Gavin Lowe. Toward a completeness result for model checking of security protocols. In *11th Computer Security Foundations Workshop Proceedings*, pages 96–105. IEEE Computer Society Press, 1998.

[17] 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.

[18] D. Maughan, M. Schertler, M. Schneider, and J. Turner. *Internet Security Association and Key Management Protocol (ISAKMP)*. IETF Network Working Group RFC 2408, November 1998.

[19] 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.

[20] Catherine Meadows. Open issues in formal methods for cryptographic protocol analysis. In *DISCEX Workshop*. DARPA, January 2000.

[21] 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.

[22] 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.

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

[24] 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.

[25] Scott Stoller. A bound on attacks on authentication protocols. Available at http://www.cs.indiana.edu/~stoller/, July 1999.

[26] Scott Stoller. A reduction for automated verification of authentication protocols. In *Workshop on Formal Methods and Security Protocols*, July 1999. Available at http://www.cs.indiana.edu/~stoller/.

[27] 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.

[28] 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.

[29] 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.

[30] 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

A.1 Terms

Assume given a set T of atomic texts and a set K of cryptographic keys disjoint from T. K is equipped with a unary inverse operator $\mathbf{inv} : \mathsf{K} \to \mathsf{K}$.

Definition A.1 A *is the algebra freely generated from* T *and* K *by the 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 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 $\mathfrak{K}$. Our assumption that A is freely generated (see also [14, 17, 22]) stretches back to Dolev and Yao [5]. Freeness is crucial for the results in this paper.

Definition A.2 *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* $a \sqsubset h$.

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 all terms* t_1.

The encryption-free *terms form the smallest set* S *including* T *and* K *and closed under concatenation. A term* t *is* simple *if it is not of the form* $g\, h$. t_0 *is a* component *of* t *if* t_0 *is simple and* $t_0 \sqsubset_\emptyset t$.

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

A.2 Strand Spaces

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.3 *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 $tr : \Sigma \to (\pm\mathsf{A})^*$.*

We will usually represent a strand space by its underlying set of strands Σ. We often ignore the distinction between signed terms and ordinary unsigned terms.

Definition A.4 Fix a strand space Σ.

1. A *node* is a pair $\langle s, i \rangle$, with $s \in \Sigma$ and i an integer satisying $1 \leq i \leq \text{length}(\text{tr}(s))$. The set of nodes is denoted by $\mathcal{N}$. If $n = \langle s, i \rangle \in \mathcal{N}$ then $\text{index}(n) = i$ and $\text{strand}(n) = s$. Define $\text{term}(n)$ to be $(\text{tr}(s))_i$, i.e. the ith signed term in the trace of s.
2. There is an edge $n_1 \to n_2$ iff $\text{term}(n_1) = +t$ and $\text{term}(n_2) = -t$ for some $t \in \mathsf{A}$. When $n_1 = \langle s, i \rangle$ and $n_2 = \langle s, i+1 \rangle$, there is an edge $n_1 \Rightarrow n_2$. We write $n' \Rightarrow^+ n$ when $n_1 = \langle s, i \rangle$ and $n_2 = \langle s, j \rangle$ for some $j > i$.
3. An unsigned term t_0 *originates* on $n \in \mathcal{N}$ iff $\text{term}(n) = +t$, $t_0 \sqsubset t$, and whenever $n' \Rightarrow^+ n$, $t_0 \not\sqsubset \text{term}(n')$.
4. An unsigned term t is *uniquely originating* iff t originates on a unique $n \in \mathcal{N}$.
5. A component t_1 of $\text{term}(n_1)$ is *new* at n_1 if, for every node n_0 such that $n_0 \Rightarrow^+ n_1$, t_1 is not a component of $\text{term}(n_0)$.

A.3 Bundles and Causal Precedence

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

Definition A.5 *Suppose $\to_{\mathcal{C}} \subset \to$; suppose $\Rightarrow_{\mathcal{C}} \subset \Rightarrow$; and suppose $\mathcal{C} = \langle \mathcal{N}_{\mathcal{C}}, (\to_{\mathcal{C}} \cup \Rightarrow_{\mathcal{C}}) \rangle$ is a subgraph of $\langle \mathcal{N}, (\to \cup \Rightarrow) \rangle$. $\mathcal{C}$ is a* bundle *if $\mathcal{N}_{\mathcal{C}}$ and $\to_{\mathcal{C}} \cup \Rightarrow_{\mathcal{C}}$ are finite, and:*

1. *If $n_2 \in \mathcal{N}_{\mathcal{C}}$ and $\text{term}(n_2)$ is negative, then there is a unique n_1 such that $n_1 \to_{\mathcal{C}} n_2$.*
2. *If $n_2 \in \mathcal{N}_{\mathcal{C}}$ and $n_1 \Rightarrow n_2$ then $n_1 \Rightarrow_{\mathcal{C}} n_2$.*
3. *$\mathcal{C}$ is acyclic.*

When a strand receives a message t, there is a unique node transmitting t from which the message was immediately received. By contrast, when a strand transmits a message t, many strands may immediately receive t.

Notational Convention A.6 *If $\mathcal{C} = \langle \mathcal{N}_{\mathcal{C}}, \to_{\mathcal{C}} \cup \Rightarrow_{\mathcal{C}} \rangle$ is a bundle, then $n \in \mathcal{C}$ means $n \in \mathcal{N}_{\mathcal{C}}$. $s \in \mathcal{C}$ means all of the nodes of s are in $\mathcal{N}_{\mathcal{C}}$.*

Definition A.7 *If $\mathcal{S}$ is a set of edges, i.e. $\mathcal{S} \subset \to \cup \Rightarrow$, 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.8 *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.4 Penetrator Strands

The actions available to the penetrator are relative to the set of keys that the penetrator knows initially. We encode this as 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:*

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

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

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

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

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

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

$\mathcal{P}_\Sigma$ is the set of all strands $s \in \Sigma$ such that $\text{tr}(s)$ 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 it is a* regular *strand or node.*

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.

Contents