Strand Space Pictures*

F. Javier Thayer Fábrega Jonathan C. Herzog
Joshua D. Guttman
The MITRE Corporation
{jt, jherzog, guttman}@mitre.org

Abstract

In this paper, we supplement the "strand space" proof methods for cryptographic protocols developed in [9, 8] by explaining another, heuristic advantage of the method. Namely, strand spaces help us draw informative pictures of cryptographic protocols, of attacks on protocols, of correctness theorems for protocols, and of crucial steps in the proofs.

As a consequence, it is possible to discover sharper characterizations of the goals of a protocol, and to discover more easily whether those goals are met.

1 Introduction

A security protocol is a convention-bound way of exchanging messages to achieve some goal such as authentication and distribution of cryptographic session keys. For purposes of analysis it is usually assumed that these messages are sent over a wide-open network, accessible also by a hostile penetrator who is able to read, alter, falsify, and delete the messages of participants.

Our previous work has focused on techniques for proving correctness conditions for protocols [8] and [9]. In this paper, we attempt to illustrate some of the geometrical intuition behind these earlier papers, mostly in the context of well-known examples of protocol failure or correctness.

A *strand* is a structure that represents a sequence of transmissions or receptions of protocol messages. They can represent either the experiences of a legitimate participant in the protocol, or one of several operations available to the penetrator. *Bundles* are structures used to define collections of strands that have a particular relation—closure under causal precedence—to a particular transmission/reception. These bundles then allow a partial order—causal dependence—to be defined.

The strand space model allows explicit and precise statements of correctness conditions using this space and relations. For example, many correctness conditions reduce to the existence of a given strand in a given bundle.

*This work was supported by the National Security Agency through US Army CECOM contract DAAB 07-96-C-E601.

We begin (Section 2) by giving a rapid overview of some of the concepts introduced in [8] and [9]. We then (Section 3) present three pictorial examples involving failure which we think clearly illustrate the strand space idea. Finally, (Section 4) we will emphasize the value of the pictorial approach by explaining its heuristic role in stating and proving correctness results.

2 Strands, Bundles and The Penetrator

A principal can either send or receive a message t; we represent this by prefixing a positive sign (transmission) or a negative sign (reception) before t. $\mathbf{A}$ denotes the set of all possible protocol messages and $(\pm\mathbf{A})^*$ the set of finite sequences of sign prefixed messages. We will denote a typical element of $(\pm\mathbf{A})^*$ by $\langle \pm, a_1, \ldots, \pm, a_n \rangle$.

Definition 2.1 *A* strand space *is a set* Σ *with a* trace *mapping* $tr : \Sigma \to (\pm\mathbf{A})^*$.

In a strand space Σ, a *node* is a pair $\langle s, i \rangle$, with $s \in \Sigma$ and i an integer satisfying $1 \leq i \leq \mathrm{length}(\mathrm{tr}(s))$. The set of nodes is denoted by $\mathcal{N}$. We will say the node $\langle s, i \rangle$ belongs to the strand s. We will refer to i as the index of $\langle s, i \rangle$. If n_1, n_2 are nodes, $n_1 \to n_2$ means $\mathrm{term}(n_1) = +a$ and $\mathrm{term}(n_2) = -a$. $n_1 \Rightarrow n_2$ means n_1, n_2 occur on the same strand with $\mathrm{index}(n_1) = \mathrm{index}(n_2) - 1$.

Definition 2.2 *Let $\mathcal{C}$ be a set of edges in a strand space, and let $\mathcal{N}_{\mathcal{C}}$ be the set of nodes incident with any edge in $\mathcal{C}$. $\mathcal{C}$ is a bundle if:*

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

We will speak of a node as being in the bundle $\mathcal{C}$ if in fact it is in $\mathcal{N}_{\mathcal{C}}$.

Bundles are closed backwards on a strand, but not necessarily forwards. Clause 2 is reminiscent of the condition characterizing message flow graphs [3]. However, strand spaces and bundles are used in a very different way from message flow graphs. The latter are intended for compile-time analysis, and are therefore a notation that characterizes all the possible execution sequences of a system. Bundles represent individual runs of a protocol. That is why bundles are required to be acyclic, while non-trivial message flow graphs contain cycles.

Given a bundle $\mathcal{C}$, an important relation on its nodes is the relation $\preceq_{\mathcal{C}}$, which holds between two nodes n_0 and n_1 just in case there is a path from n_0 to n_1 along edges in $\mathcal{C}$. The relation $\preceq_{\mathcal{C}}$ expresses the fact that one node is a causal predecessor of another. When $\mathcal{C}$ is clear, we write simply $\preceq$.

Finally the model contains a special set of strands which represent the actions available to a hostile penetrator. These actions include the abilities to

$$
\begin{tikzcd}
+a \arrow[rr] \arrow[dddd, Rightarrow] & & -a \arrow[d, Rightarrow] & +b \arrow[d, Rightarrow] \\
& -c \arrow[dd, Rightarrow] & +c \arrow[l] \arrow[r] \arrow[d, Rightarrow] & -c \arrow[d, Rightarrow] \\
& & -d \arrow[d, Rightarrow] & +d \arrow[l] \\
& +e \arrow[r] & -e \arrow[d, Rightarrow] & \\
-f & & +f \arrow[ll] \arrow[d, Rightarrow] & \\
& & +g &
\end{tikzcd}
$$

Figure 1: A Bundle

1. Concatenate, decompose, repeat, or discard messages.
2. Utter a key known to the penetrator from the start.
3. Utter a text value (e.g. a name, a new nonce).
4. Encrypt or decrypt a message using a previously obtained key.

It is also possible to extend the set of penetrator traces given here if it is desired to model some special ability of the penetrator, such as the ability to cryptanalyze some kinds of encrypted messages, without any essential change to our overall framework.

3 Pictures

Mathematically, bundles provide a way of "piecing together" individual strands in a way that preserves causality. Figure 1 shows how a bundle pieces together various strands. In this example, every node in the strand space belongs to the bundle. It thus follows that the backward closure property (that is 3 of Definition 2.2) is satisfied. The reader can easily verify that the graph is acyclic.

Notice that in this example, positive nodes can have multiple out-arrows of type $\rightarrow$, but only one of type $\Rightarrow$. Also, one positive node has no out-arrows at all, because it is the last on the strand and no other node receives the message it sends.

Though the picture in Figure 1 represents a legitimate bundle, the corresponding strand space does not represent a meaningful protocol. In the examples that follow, we consider examples which are more directly related to protocol analysis.

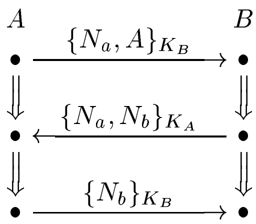


Figure 2: Needham-Schroeder

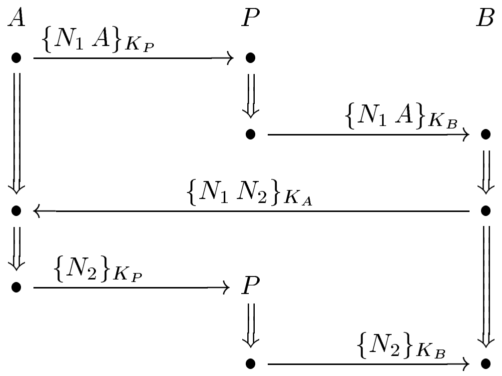


Figure 3: Needham-Schroeder Infiltrated

3.1 Needham-Schroeder

This protocol was proposed by Needham and Schroeder [7]. In the form we discuss it here, we assume that each participant has somehow acquired the other's public key. The intended result of this protocol is that the two participants should come to share access to the values N_a and N_b, each associating these values with the other participant, and no other party should be in possession of them. The protocol might be used in a context where the two values are hashed together to yield a shared symmetric key for an encrypted session, for instance.

Figure 2 illustrates the naive view of how the original Needham-Schroeder protocol message exchange works. In this view, the typical bundle consists of two strands pieced together in a complementary way. Were this geometrical picture always accurate, then the Needham-Schroeder protocol would indeed provide mutual authentication.

However, Gavin Lowe has shown this protocol is flawed [4]. Lowe constructs an example run in which B thinks he shares nonces N_1, N_2 only with A, but P knows these values, and moreover A has no matching run with B. Geometrically (Figure 3) the failure can be understood by the existence of a bundle consisting of non-complementary strands constructed by the insertion of penetrator strands.

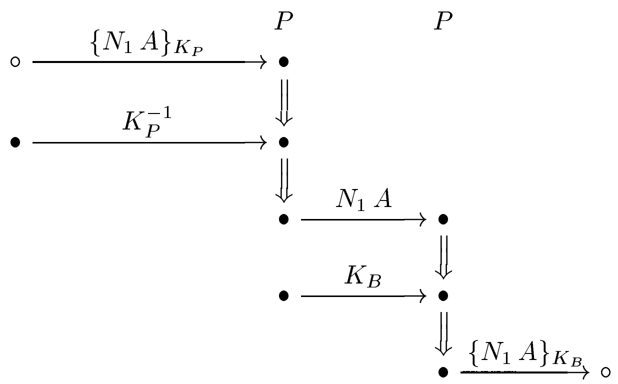


Figure 4: Needham-Schroeder: Penetrator's First Step

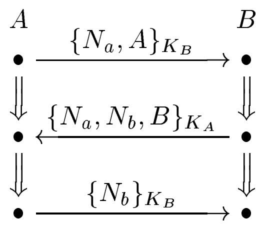


Figure 5: Needham-Schroeder-Lowe

Figure 3 is a simplification because it does not show an analysis of the penetrator's activity in terms of the primitive types of penetrator activity as summarized in Section 2. An explicit version of the penetrator's first action is shown in Figure 4. The open circles ($\circ$) show the two points at which this diagram meshes with the first nodes of A and B's strands at the top of Figure 3.

The Needham-Schroeder protocol may be fixed with a slight change, also due to Lowe [5]. All that is needed is for the responder to include his name in the second message. The corrected Needham-Schroeder-Lowe protocol [5] is illustrated in Figure 5.

3.2 Woo-Lam

The Woo-Lam protocol [10, 2] uses symmetric keys shared between the principals and a trusted authentication server. It is intended to allow the initiator A to authenticate himself to the respondent B with the help of the trusted server S, which is assumed to share a long-term key with each of them. After A indicates his desire to authenticate himself, B issues a nonce that serves as a challenge. A encrypts this with his long-term key K_{AS}, shared with S. B

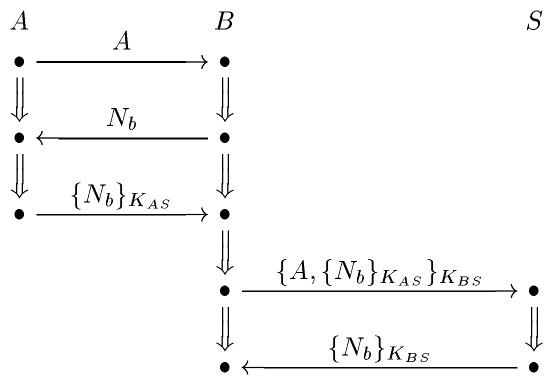


Figure 6: Woo-Lam

forwards the resulting value to S, indicating who it is intended to authenticate. S retrieves that principal's long-term key and decrypts the value. If this is the same key that was used to encrypt the challenge, then the challenge will be recovered, while otherwise garbage will be produced. In either case, the result is sent to B, encrypted in B's key. B simply checks whether the value contained in the final message is the challenge or garbage.

It is known that this protocol is vulnerable. An attack is shown in Figure 7. In this attack, the penetrator waits until B wants to initiate a session (with no matter what responder). The penetrator then initiates a session (purporting to be some third party A) with B as responder. In this attack, B as initiator dupes himself as responder by encrypting his own challenge. The strand space picture makes the situation particularly clear by showing B's actions in both roles, and just how they are interleaved by the mischievous penetrator.

3.3 Otway-Rees

The final example we consider is a vulnerability of the Otway-Rees protocol discussed in [8]. The goal of the protocol is to mutually authenticate initiator and responder and to distribute a session key generated by a server, as is illustrated by Figure 8. As is shown in [8], this protocol does achieve mutually authentication under certain conditions. Nevertheless, there is no guarantee that the participants receive the same session key from the server. (See Figure 9.)

4 General Pictures

In the previous section, we gave examples of how strand space pictures can be used to show normal protocol runs (Figures 2, 6, 8), and of how they may be used to show attacks (Figures 3, 7, 9). In each of these cases, the picture represents a single situation, which is either an *example* of the desired behavior

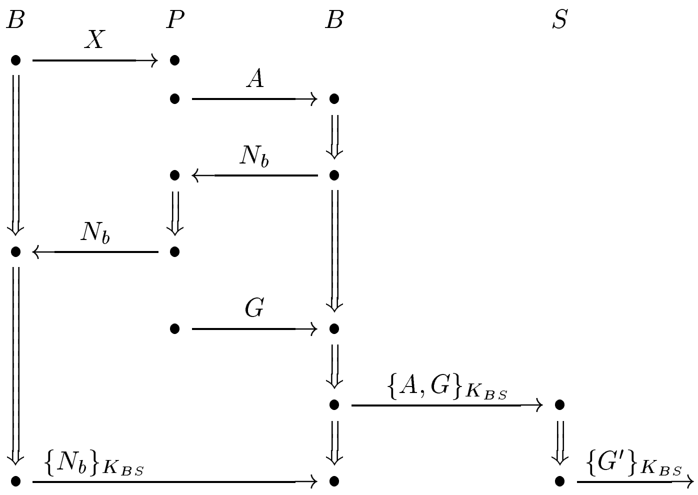
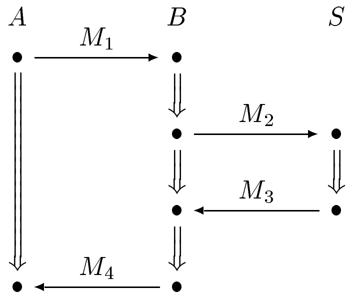


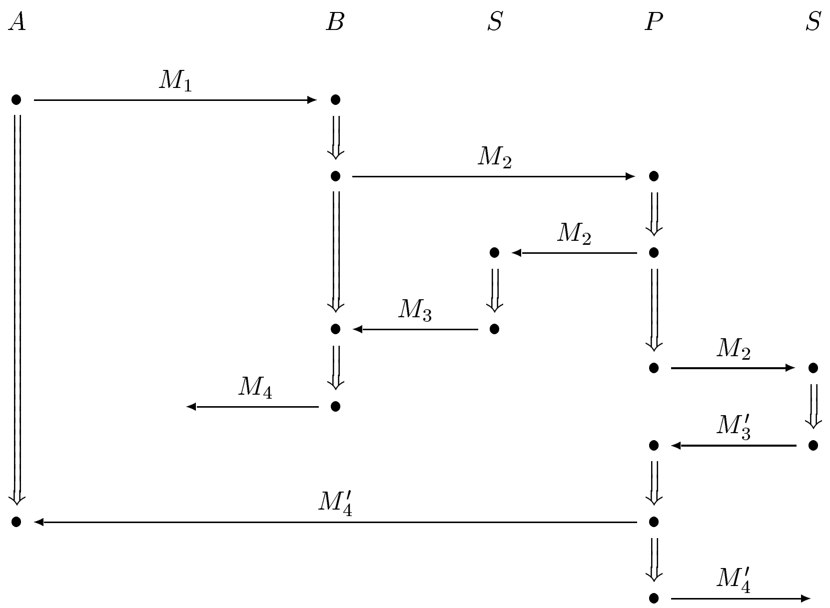
Figure 7: Woo-Lam Infiltrated



Where

1. $M_1 = M\,A\,B\,\{N_a\,M\,A\,B\}_{K_{AS}}$.
2. $M_2 = M\,A\,B\,\{N_a\,M\,A\,B\}_{K_{AS}}\,\{N_b\,M\,A\,B\}_{K_{BS}}$.
3. $M_3 = M\,\{N_a\,K_{AB}\}_{K_{AS}}\,\{N_b\,K_{AB}\}_{K_{BS}}$.
4. $M_4 = M\,\{N_a\,K_{AB}\}_{K_{AS}}$.

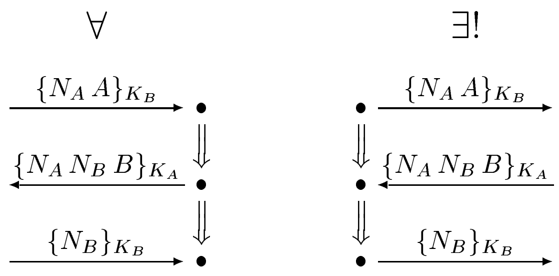
Figure 8: Otway-Rees



Where

1. $M_1 = M\,A\,B\,\{N_a\,M\,A\,B\}_{K_{AS}}$.
2. $M_2 = M\,A\,B\,\{N_a\,M\,A\,B\}_{K_{AS}}\,\{N_b\,M\,A\,B\}_{K_{BS}}$.
3. $M_3 = M\,\{N_a\,K_{AB}\}_{K_{AS}}\,\{N_b\,K_{AB}\}_{K_{BS}}$.
4. $M'_3 = M\,\{N_a\,K'_{AB}\}_{K_{AS}}\,\{N_b\,K'_{AB}\}_{K_{BS}}$.
5. $M_4 = M\,\{N_a\,K_{AB}\}_{K_{AS}}$.
6. $M'_4 = M\,\{N_a\,K'_{AB}\}_{K_{AS}}$.

Figure 9: Otway-Rees Infiltrated



Assuming:

1. $K_A^{-1} \notin \mathbf{K}_{\mathcal{P}}$;
2. $N_a \neq N_b$ and N_b is uniquely originating in Σ.

Figure 10: Needham-Schroeder-Lowe: One Authentication Goal

of the protocol, or a *counterexample* to the conjecture that the protocol achieves an authentication goal.

We will now argue that they may also be used in two other ways:

1. They may be used to represent proof goals; and
2. They may be used to express general assertions, for instance as arise in the course of proofs.

These two heuristic roles are highly valuable when humans are trying to state exact correctness guarantees for protocols, and when they are trying to develop simple and convincing proofs of those guarantees.

4.1 Pictures of Proof Goals

Pictures showing portions of a bundle can be used to convey an intuitive grasp of proof goals. For example, authentication guarantees can frequently be understood as agreement properties (see [6]; cp. [11]), which state that when one principal has undergone its portion of a protocol run, then an interlocutor has undergone a "matching" portion. More or less stringent specification result from varying senses of "matching."

Agreement properties are very easily formalized in the strand space framework; they stipulate that when a strand belongs to a bundle, then there also exists a "matching" strand in the same bundle [9]. Such assertions are easily displayed as a diagram with a few supplementary indications, for instance if the relevant notion of "matching" is that the expected strand should exist uniquely in the bundle. Figure 10 illustrates this in the case of one of the Needham-Schroeder-Lowe authentication guarantees [9].

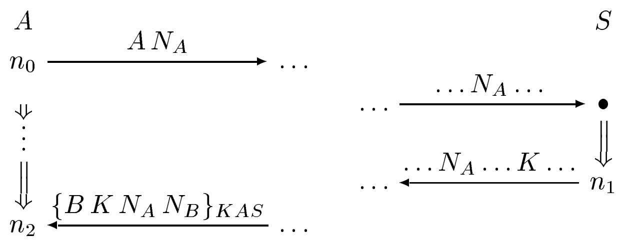


Figure 11: Yahalom: A Freshness Goal

As another example of a proof goal, freshness properties can also be easily formalized in the strand space model. For example, Figure 11 shows a portion of a bundle for the Yahalom protocol [1]. In this case, a participant A creates a new nonce N_A at node n_0 and later, at n_2, receives the message $\{B\,K\,N_A\,N_B\}_{K_{AS}}$. A wishes to be sure that the session key K, generated by a server at node n_1, is fresh. As Figure 11 shows, the freshness goal is the fact that $n_0 \preceq n_1 \preceq n_2$, i.e. that there are paths from n_0 to n_1 and from n_1 to n_2 along edges in the bundle.

4.2 Pictures of Proofs

We have found that pictures play a valuable role in developing and presenting strand space proofs. As an example, let us again consider the authentication guarantee shown in Figure 10.

The key step in this proof is to consider what strand could be responsible for first emitting a message in which N_b occurs, but not in the context $\{N_a\,N_b\,A\}_{K_A}$. Subject to the assumptions that N_b is freshly emitted by B in $\{N_a\,N_b\,A\}_{K_A}$ and that K_A^{-1} is uncompromised, it follows ([9, Lemmas 4.4 and 4.6]) that there is a non-penetrator strand as shown on the left side of Figure 12. This strand is easily shown to complete to the form required in Figure 10. Thus, illustrations such as 12 turn out to be a powerful tool to capture intuition in the course of developing a proof, or to convey intuition to a reader in an exposition of a proof. The figures can play a role similar to the role figures play in proofs in elementary plane geometry. There is, however, the fact that the precise logical content of a bundle diagram is easy to make precise, using Definition 2.2, whereas the precise meaning of diagrams in geometry was not understood from the time of Euclid until the end of the nineteenth century.

5 Conclusion

Formal methods are sometimes criticized for being too arcane; those who do not understand them very well sometimes think that they inhibit understanding and make it harder to explore issues with the benefit of a vivid and well-trained intuition. Probably this is sometimes true. Moreover, in many cases, the benefits

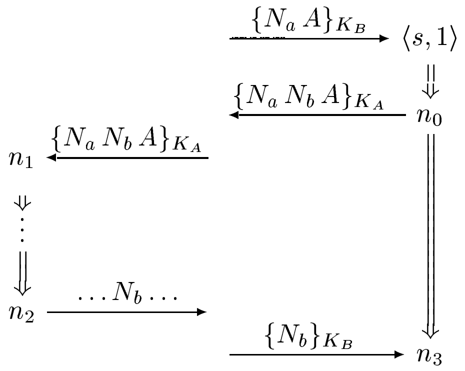


Figure 12: Proving an Authentication Guarantee

from existing automated support tools do not outweigh the loss of intuition.

In this paper we have tried to show—by means of one example—that this is not always the case. We believe that the strand space formalism is particularly clean and simple. It identifies a few basic concepts, together with some corresponding methods of reasoning. In addition, it offers a concrete, suggestive way of drawing diagrams; they help even a relatively naïve user to explore the different things that may go wrong in a cryptographic protocol, or the different reasons why it may in fact go right. We hope it can serve as a model of how formal methods can combine rigor with an increase in the heuristic power that an analyst has at his disposal.

Acknowledgements. We are grateful to Sylvan Pinsky and his colleagues at NSA for support, encouragement, and discussions.

References

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

[2] John Clark and Jeremy Jacob. A Library of Security Protocols with Attacks. Technical report, Dept. of Computer Science, University of York, June 1995.

[3] Peter B. Ladkin and Stefan Leue. Interpreting message flow graphs. *Formal Aspects of Computing*, 3(1), 1994.

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

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

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

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

[8] F. Javier Thayer Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Honest ideals on strand spaces. In *Proceedings of the 11th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, June 1998.

[9] F. Javier Thayer Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Why is a security protocol correct? In *1998 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 1998.

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

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