Adding Time to a Logic of Authentication

Paul F. Syverson
Code 5543
Naval Research Laboratory
Washington, DC 20375
(syverson@itd.nrl.navy.mil)

Abstract: In [BAN89] Burrows, Abadi, and Needham presented a logic (BAN) for analyzing cryptographic protocols in terms of belief. This logic is quite useful in uncovering flaws in protocols; however, it also has produced confusion and controversy. Much of the confusion was cleared up when Abadi and Tuttle provided a semantics for a version of that logic (AT) in [AT91].

In this paper we present a protocol to show that both BAN and AT are not expressive enough to capture all of the kinds of flaws that appear to be within their scope. We then present a logic that adds temporal formalisms to AT and that is rich enough to reveal the flaws in the presented protocol; nonetheless, this logic is sound with respect to the same semantics that was given in [AT91]. Finally, we argue that any approach of this type is inadequate by itself to demonstrate the absence of such flaws. We must supplement the formal logic with semantic analysis techniques.

1 Introduction

This paper presents a class of attacks on cryptographic protocols that are not representable using BAN or AT. We also show how incorporating into the syntax of AT features already present in the Abadi-Tuttle semantics allows us to represent such flaws. (We assume the reader has an elementary understanding of cryptographic protocols and associated concepts, e.g., key distribution, authentication, etc. We are concerned with the security of the protocols rather than the cryptography they employ. Thus, e.g., questions about the release of secrets by direct cryptanalysis of ciphertext are not addressed here.)

This is not the first paper to discuss such attacks. Similar attacks have been described in [BGH$^+$92], [DvOW92], [Sne92], and [Syv]. Nor is this the first paper to discuss logical solutions to such attacks. In [Sne92], Snekkenes uses Bieber's logic CKT5 [Bie90] to analyze a similar attack. Nonetheless, the results herein are significant for a number of reasons:

(1) For good or ill, BAN has become the clear favorite as a formal method for cryptographic protocol analysis; there have been numerous publications that make use or misuse of BAN. The oft cited reason for this is BAN's simplicity. We have elsewhere discussed the dangers lurking behind this apparent simplicity ([Syv91], [Syv92]) and will not rehash them here. However, given that this remains a compelling force and that people will continue in their desire to use BAN, it behooves us to produce a version of BAN that will make such usage as reasonable as possible without compromising the apparent simplicity that motivates it. The logic presented herein does that.

(2) Most of the work in this direction has already been done by Abadi and Tuttle. By producing a possible world semantics based on a reasonable model of computation they allow users of their logic to attach meanings to expressions that are both intuitive and mathematically precise. This removes much of the confusion and ambiguity that has hitherto been associated with the 'idealization' of protocols, as it has come to be called. It also allows one to give semantic proofs, though Abadi and Tuttle did not present any. One example was presented in [Syv92], and another is presented in this paper.

(3) We must recall that AT is not the same as BAN; it is more expressive. For example, it can represent the possession of a key in addition to belief in its goodness as a key.[1] One might conclude that, as an extension, AT gives away some of the simplicity associated with the original BAN, but just the opposite is true. Because the changes that yielded AT are based on a detailed semantic model, one has a much more unified picture than emerges from a mere collection of syntactic rules—no matter how intuitively reasonable those rules may be. Developed differences of AT from BAN are primarily there to make the logic more compatible with a standard model theoretic semantics. Thus, in our experience AT is actually conceptually simpler to use than BAN.

Though the logic presented in this paper is more expressive than AT, it is sound with respect to the same model of computation as was presented in [AT91]. Thus, it allows us to reason about more kinds of attacks, but the intuitive complexity is all but unchanged. It allows us to reason about the protocol attacks cited above. And, it allows us to do so in a version of AT. And, as we have argued, AT is itself the most intuitively clear available version of BAN, the formal method of choice for most people.

(4) Snekkenes showed in [Sne92], that one can give BAN proofs in effect sanctioning protocols subject to the type of attack we will be discussing. We strengthen his arguments by showing that the threat is even greater than he demonstrated. First, the attack analyzed herein is the result of penetrator initiated protocol runs. The attack in [Sne92] requires the penetrator to intercede in a protocol initiated

This paper will appear appear in *Proceedings of the First ACM Conference on Computer and Communications Security*, Fairfax, Virginia, November, 1993.

---

[1] As far as I know, the first BAN-like logic to add this particular expressive capability was that of [GNY90].

by an honest principal. Second, the attack in [Sne92] involves the substitution of a principal name for a session key. Our attack involves the substitution of nonce for a session key, which is conceivably not as likely to be noticed since it is the substitution of one freshly generated random number for another. (A nonce is a random number typically generated by a principal and used by him to determine the freshness of messages containing it.) The attack presented herein is also successful under a broader class of implementation assumptions than, e.g., the one in [Syv]; nonetheless, the one herein and the one in [Sne92] are attacks on artificial examples while the attack in [Syv] is on a published protocol.

The remainder of the paper will be set out as follows. In the next section we present a key distribution protocol that has no flaws representable in BAN or AT and show how to attack it. In §3 we add a temporal operator to AT and give its semantics as well as corresponding axioms. In §4 we discuss the nature of the flaw in the protocol given in §2 and show how to use the expanded logic to represent it. Finally, we present some conclusions and discuss some of the limitations on the logical analysis of such attacks.

2 A Key Distribution Protocol

In this section we present a two-party key distribution protocol. It is possible to prove in AT that both principals believe they have a good session key for talking to each other and that they both believe the other believes this as well. A similar proof is also possible in BAN. These are the usual goals that one would try to verify using BAN, though what one attempts to prove ultimately depends on what one intends the protocol to accomplish.[2]

Two-party Key Distribution Protocol

$A \rightarrow B$: A, N_a
$B \rightarrow S$: $B, N_b, \{N_b, N_a, A\}_{K_{bs}}$
$S \rightarrow A$: $S, B, N_s, \{N_a, N_s, B\}_{K_{as}}$
$A \rightarrow S$: $A, \{K_{ab}, N_s, B\}_{K_{as}}$
$S \rightarrow B$: $S, \{N_b, K_{ab}, A\}_{K_{bs}}$
$B \rightarrow A$: $\{N_a + 1\}_{K_{ab}}$

Here A and B are principals, and S is an authentication server. The arrow ($\rightarrow$) indicates the sending of a message. N_x is a nonce generated by X, and K_{xy} is a key used only by X and Y. Upon receipt of N_a in the third message, A knows that she has a fresh message from S and that, given the protocol format, S has received N_a from B. Thus, S and B are authenticated to A. For the fourth message, A generates the session key and encrypts it along with N_s and B using K_{as} to authenticate herself to S and to pass on the key for B. S then sends the key to B encrypted together with N_b and A using K_{bs}. This authenticates S to B and lets B know that the key is freshly from A. The last message lets A know that B has the key.

2.1 Attacking the Protocol

In this section we show how the protocol given above is subject to attack by substituting a random number generated as a nonce for one generated as a session key. We use 'E' to refer to a penetrator, and 'E_x' to indicate that she is masquerading as X. (If the penetrator intercepts a message, we assume that she not only receives it but also prevents its passage to the intended party.)

$E_a \rightarrow B$: A, N_a
$B \rightarrow S$: $B, N_b, \{N_b, N_a, A\}_{K_{bs}}$
$S \rightarrow E_a$: $S, B, N_s, \{N_a, N_s, B\}_{K_{as}}$

$E_b \rightarrow A$: B, N_s
$A \rightarrow E_s$: $A, N'_a, \{N'_a, N_s, B\}_{K_{as}}$

$E_a \rightarrow S$: $A, \{N'_a(= K_{ab}), N_s, B\}_{K_{as}}$
$S \rightarrow B$: $S, \{N_b, N'_a(= K_{ab}), B\}_{K_{bs}}$
$B \rightarrow E_a$: $\{N_a + 1\}_{K_{ab}}$

Here is how the attack works: for the first three steps of the protocol things proceed normally except that E is masquerading as A. At that point E also begins the same protocol with A, this time masquerading as B. She also intercepts the message that A attempts to send to S in the second step of that protocol. (After that the latter protocol round is dropped.) She uses this same message (minus the plaintext N'_a) as the fourth message in the original protocol. Thus, in this message, in the place of the session key is A's nonce N'_a. The server then passes the supposed session key on to B. Because this appeared as a plaintext nonce in the other protocol run, E has succeeded in starting a session with B in which she has successfully set herself up as A and obtained the session key.

We call such an attack a *causal consistency attack* because it is possible for all the participants to faithfully execute the protocol and yet not have consistent records of the message history. Similar attacks have been discussed in [BGH$^+$92], [Sne92], and [DvOW92]. In fact the above protocol and attack was inspired by a draft version of a proposed ISO pure authentication protocol that was shown to be flawed in [DvOW92]. Though never actually proposed for use, the above attack is somewhat more disturbing because it involves the distribution of a bad session key as well as a spoofed authentication. Using the formalism we introduce presently, we will show how to represent such attacks.

3 Adding time to AT

There are various ways we might rule out this attack. One way would be to assume that the system is structured so that principals can always recognize distinct types within a message field. This is probably the simplest solution both to state and to implement. But, there may be problems. In the syntax of AT, nonces do not have a type. There is nothing to rule out using keys to guarantee message freshness. Thus, even though keys do have a type, their use as nonces is not precluded.

Ultimately we do not rule out such attacks simply by having a logic that is typed strongly enough to assume them away. All we do thereby is assume that the the attack is handled in some other part of our security analysis. It is perfectly legitimate to say that BAN and AT were not meant to deal with such kinds of attacks—although, if this be the case it would be better if it were explicitly stated, and typing should be made strong enough to support the claim. That way the assumptions are consistent with the notation and clearly set out. And, of course some assumptions are inevitable; for example, all formalisms for analyzing cryptographic protocols explicitly assume perfect encryption. Nonetheless, the

---

[2] Cf. [Syv91] and [Syv] for a discussion of these issues. Also cf. [BAN89] and [BAN91], which also appears as an added note in the revised version of [BAN89].

assumption of other security mechanisms may not always help. For example, direction bits are a standard way to deal with attacks that involve the replay of part of one message in another message in the same protocol. This might preclude some attacks, but the above example shows that they cannot serve as a general solution since they are of no help here. Also, it is conceivable that some causal consistency attacks involve only substitutions of terms of the same type, for example, an old key for a new one. Thus, even if the assumption that principals can recognize types in message fields is simple to state and to implement, it might be advisable to have alternatives available. Another possibility would be to introduce in effect a more general notion of types for messages. Each encrypted section of a message could contain a protocol identifier and a message/section number (saying, e.g., "This is part of message 3 in a run of Kerberos"). Such fields are not unreasonable in practice.[3] This probably rules out all such attacks, and it would not be difficult to add these fields in a protocol implementation; however, they would add to the computation and communication costs of each message whether or not they are needed. This additional expense might be outweighed by the cost of determining where the fix is needed and where it isn't. A more immediate practical concern is the redundancy introduced by this solution. For example, it would allow for exhaustive search attacks on password based encryption as in, e.g., Kerberos.

Fortunately the semantics of AT is already robust enough to indicate the flaw in the above protocol. We need only add the corresponding syntax to reflect this in the logic. The semantics of AT is a possible world semantics along the lines of [HM90]. The truth values of logical formulae are determined by the state of the distributed system at a specific time. Each principal is assumed to have a local state which may change when someone performs an action. A global state is a set of local states, one for each principal plus one for the environment. A run is just an infinite sequence of global states. The local state of any principal includes the history of all actions the principal has performed up to that time in that run and a set of keys available to that principal. The global history includes a key set, all local histories, plus one buffer for each principal—to handle messages that were sent but not yet received. The complete details need not concern us. The important point is that the truth values of formulae in the logic are fixed by specifying a run and a time. This makes it fairly easy to add syntax from temporal logic to AT.

3.1 Temporal Axioms

We now introduce axioms to reason about time. Intuitively '$\boxminus$' means at all points in the run prior to the current one, and '$\diamondminus$' means at some point in the run prior to the current one. These are interchangeable according to the definition: $\boxminus\varphi \leftrightarrow \neg\diamondminus\neg\varphi$ (for any formula φ). Given formulae φ and ψ the axioms are as follows[4]:

K $\boxminus(\varphi \supset \psi) \supset (\boxminus\varphi \supset \boxminus\psi)$

4 $\boxminus\varphi \supset \boxminus\boxminus\varphi$

D $\boxminus\varphi \supset \diamondminus\varphi$

L $\boxminus(\varphi \wedge \boxminus\varphi \supset \psi) \vee \boxminus(\psi \wedge \boxminus\psi \supset \varphi)$

Z $\boxminus(\boxminus\varphi \supset \varphi) \supset (\boxminus\diamondminus\varphi \supset \boxminus\varphi)$

We also need to add one rule to the logic. It says that if a formula φ is a theorem of the logic, then so is $\boxminus\varphi$.

Nec From $\vdash \varphi$ infer $\vdash \boxminus\varphi$.

K and **Nec** make sure that we have a normal modal logic. (Cf. [Gol92]) These basically guarantee that the temporal operators behave reasonably with respect to the rest of the logic. Each of the other axioms captures a feature of time that we desire. **4** gets us transitivity. (Something that was always true in the past was always true at any time prior to any time in the past.) **D** guarantees that we don't run out of time points (seriality). **L** guarantees that all points in time are connected. And, **Z** guarantees that time is discrete. (Between any two points in time there are at most finitely many other points.)

We need to give conditions for $\diamondminus\varphi$ to be true at a point (r, k), where r is a run and k is a time. We write this '$(r, k) \models \diamondminus\varphi$'. (Note that, because of their interdefinability, this also gives us the conditions for $(r, k) \models \boxminus\varphi$.) Given a formula, φ, a run r, and a time, k:

$$(r, k) \models \diamondminus\varphi \text{ iff } (r, k') \models \varphi \text{ for some } k' < k.$$

In practice we may not even need the full logic that results from adding the above to AT. The applications we envision require no interplay of temporal and doxastic (belief) modalities. Thus, we can restrict the application of doxastic operators to formulae free of temporal operators and the application of temporal operators to formulae free of doxastic operators and vice versa. The resulting logic is sound with respect to the semantics given in [AT91] once the truth conditions just given are added. Should it be necessary in some application for us to allow formulae in which the two types of modalities are applied successively, we can then make use of the full logic. It is fairly straightforward to show that this logic is also sound with respect to the given semantics.

4 Exposing Protocol Flaws Using the Logic

In order to logically represent the flaw in the key distribution protocol given above we must state some requirement in the logic and then show that the protocol fails to meet it. In effect, the temporal operators allow us to use the protocol to specify its own requirements. We simply require that each principal sees messages and sends them in the right order. This amounts to requiring each principal's history of the protocol to match the others', as far as they go. (We cannot require that they match entirely because we don't want to deem protocols flawed if they fail to complete. Even if they do run to completion, there is no way for the sender of the last message to know that the intended receiver in fact received it.) This approach to security requirements is basically the same as that in [DvOW92], where matching histories was given as a condition in the definition of 'secure protocol'. However, we will go further and connect this with protocol analysis in a formal language. We will present the requirements in two sections. First, we present the causal requirements. We call these 'causal requirements' because they specify the only acceptable cause of a principal's seeing a received message, viz: the appropriate principal sent that message at some previous time.

---

[3] This suggestion was given to me by Martín Abadi.

[4] These are standard axioms for discrete (past) time. The axioms and rule have their historical names. Cf. [Gol92].

We can state the causal requirements easily by reading them off the protocol in reverse order. (We ignore the last message since it plays no role in the attack.) For perspicuity we present a list of requirements rather than a single large requirement.

Causal Requirements

1. $(B\ \mathit{sees}\ (S, \{N_b, K_{ab}, A\}_{K_{bs}})) \supset$
   $\diamondminus(S\ \mathit{said}\ (S, \{N_b, K_{ab}, A\}_{K_{bs}}))$
2. $(S\ \mathit{sees}\ (A, \{K_{ab}, N_s, B\}_{K_{as}})) \supset$
   $\diamondminus(A\ \mathit{said}\ (A, \{K_{ab}, N_s, B\}_{K_{as}}))$
3. $(A\ \mathit{sees}\ (S, B, N_s, \{N_a, N_s, B\}_{K_{as}})) \supset$
   $\diamondminus(S\ \mathit{said}\ (S, B, N_s, \{N_a, N_s, B\}_{K_{as}}))$
4. $(S\ \mathit{sees}\ (B, N_b, \{N_b, N_a, A\}_{K_{bs}})) \supset$
   $\diamondminus(B\ \mathit{said}\ (B, N_b, \{N_b, N_a, A\}_{K_{bs}}))$
5. $(B\ \mathit{sees}\ (A, N_a)) \supset \diamondminus(A\ \mathit{said}\ (A, N_a))$

Of course we don't expect the above formulae to be valid, i.e., true in all possible runs. For example, we can't expect $(B\ \mathit{sees}\ (A, N_a)) \supset \diamondminus(A\ \mathit{said}\ (A, N_a))$ to be valid since anybody might send (A, N_a) to B. We only expect the assumptions to hold in those runs where all the protocol participants faithfully execute their part of the protocol. We need to formally stipulate these faithfulness assumptions. Once we have set this out, security will amount to the claim that any run that is faithful satisfies the above requirements. In order to be faithful, participants should only proceed if they have seen the message they were to have received most recently.[5] We also need to address that faithfulness of this protocol is not well-founded. The problem is that anyone could start the protocol, and, if participants looked only at the last message they were to have received when deciding if they should proceed, the protocol could still be faithfully executed. Thus, we also require that no participant can be faithful if he picks the protocol up in the middle and executes faithfully from that point on. In particular, A should not be willing to send the fourth message if she did not send the first, and S should not be willing to send the fifth message if he did not send the third. (Encryption guarantees that he said at least the last part of the message, but this has nothing to do with his faithfulness.) Our faithfulness assumptions are thus the following:

Faithfulness Assumptions

1. $(B\ \mathit{said}\ (B, N_b, \{N_b, N_a, A\}_{K_{bs}})) \supset$
   $\diamondminus(B\ \mathit{sees}\ (A, N_a))$
2. $(S\ \mathit{said}\ (S, B, N_s, \{N_a, N_s, B\}_{K_{as}})) \supset$
   $\diamondminus(S\ \mathit{sees}\ (B, N_b, \{N_b, N_a, A\}_{K_{bs}}))$
3. $(A\ \mathit{said}\ (A, \{K_{ab}, N_s, B\}_{K_{as}})) \supset$
   $\diamondminus(A\ \mathit{sees}\ (S, B, N_s, \{N_a, N_s, B\}_{K_{as}}))$
4. $(A\ \mathit{said}\ (A, \{K_{ab}, N_s, B\}_{K_{as}})) \supset$
   $\diamondminus(A\ \mathit{says}\ (A, N_a))$
5. $(S\ \mathit{said}\ (S, \{N_b, K_{ab}, A\}_{K_{bs}})) \supset$
   $\diamondminus(S\ \mathit{sees}\ (A, \{K_{ab}, N_s, B\}_{K_{as}}))$
6. $(S\ \mathit{said}\ (S, \{N_b, K_{ab}, A\}_{K_{bs}})) \supset$
   $\diamondminus(S\ \mathit{says}\ (S, B, N_s, \{N_a, N_s, B\}_{K_{as}}))$

We represent the conjunction of the above causal requirements as **CR** and the conjunction of the above faithfulness assumptions as **FA**. We can then state our criterion for causal consistency (**CCC**) as the requirement that the following formula be valid:

$$\mathbf{FA} \supset \mathbf{CR}$$

Since we have seen that the causal consistency of the protocol from which **CR** and **FA** were derived is subject to attack, we should expect this formula to be invalid. In order to demonstrate that it is invalid, we need simply find a run r and time k such that $(r, k) \models \mathbf{FA}$ but $(r, k) \not\models \mathbf{CR}$. We will demonstrate invalidity using the very attack we set out above to determine r. If there is a time during r that will show that **CCC** is invalid, it is the time k in the attack when A says $(A, N_a', \{N_a', N_s, B\}_{K_{as}})$.

The truth of **CCC** at (r, k) depends on whether or not A also said $(A, \{K_{ab}, N_s, B\}_{K_{as}})$ at (r, k). For if, $(r, k) \models A\ \mathit{said}\ (A, \{K_{ab}, N_s, B\}_{K_{as}})$, then we violate Faithfulness Assumption 3 in virtue of the fact that A has never seen $S, B, N_s, \{N_a, N_s, B\}_{K_{as}}$ when she says $(A, \{K_{ab}, N_s, B\}_{K_{as}})$. This would make **CCC** true at (r, k) by virtue of a false antecedent. Thus, we need to decide whether or not $(r, k) \models A\ \mathit{said}\ (A, \{K_{ab}, N_s, B\}_{K_{as}})$. In determining what was said by a principal we must respect the actual syntax that was used (and its semantic interpretation). And, there is nothing in either the syntax or semantics of AT that allows us to equate two message components given by distinct terms simply because they have the same bit representation. Thus, $(r, k) \not\models A\ \mathit{said}\ (A, \{K_{ab}, N_s, B\}_{K_{as}})$.

Once we have established this, the invalidity of **CCC** quickly follows. All of **FA** is true at (r, k) but Causal Requirement 2 is false because S sees $(A, \{K_{ab}, N_s, B\}_{K_{as}})$ without A's having said it first. (Since the logic is sound this also amounts to a proof that **CCC** is not a theorem. This is only significant, however, if there is a special premium placed on being strictly formal.)

5 Conclusions

We have shown that by incorporating temporal logic into AT, it is possible to logically demonstrate that protocols subject to causal consistency attacks are flawed. It seems that we thus have a technique for establishing the security of cryptographic protocols with respect to causal consistency; however, we should not be overly enthusiastic about these results by themselves. We have shown that a syntactic security requirement is invalid. But, how would we go about showing that the requirement is valid for a given protocol? We could try to produce a logical derivation, but this is highly unlikely in all but trivial cases. The only alternative is to show that the requirement holds in all possible runs of the system. Here formal proof is no help, and our failure to find a counterexample does not imply that there isn't one. In practice, what this means is that we have the ability to represent such flaws but not as yet the ability to detect them or to demonstrate their absence.

What we need is a tool to aid us in semantic analysis, to provide assurances (if not proof) that a protocol is secure in this sense. In other words, we need a model checker, perhaps along the lines of [CES86]. Such work has recently

---

[5] I know that I am skirting quite close to the very slippery notion of honesty. (Cf. [FH88] and [AT91] for a discussion of some of the issues.) However, I do not need such subtlety to require that participants be faithful.

been applied to cryptographic protocol analysis. In [SM93] we present a temporal language for reasoning about cryptographic protocol requirements for which the NRL Protocol Analyzer ([Mea91], [Mea92]) serves as a semantic model checker. This tool has already demonstrated its usefulness by finding flaws in published protocols. Nonetheless, one of the major appeals of logics such as BAN and AT has been their apparent simplicity. It would be nice if we could continue to use them for the analyses to which they are suited and could then supplement this with a semantic analysis of causal consistency for the same logic. It is hoped that this work provides a step in that direction.

Acknowledgements

I thank the anonymous referees for helpful comments on a draft of this paper. I offer similar thanks to the following people: Martín Abadi, Jim Gray, John McLean, Cathy Meadows, Paul van Oorschot, Åsmund Skomedal, and Einar Snekkenes.

References

[AT91] Martín Abadi and Mark R. Tuttle. A Semantics for a Logic of Authentication. In *Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing*, pages 201–216. ACM Press, August 1991.

[BAN89] Michael Burrows, Martín Abadi, and Roger Needham. A Logic of Authentication. Research Report 39, Digital Systems Research Center, February 1989. Parts and versions of this material have been presented in many places including *ACM Transactions on Computer Systems*, 8(1): 18–36, Feb. 1990, and *Proceedings of the Royal Society of London A*, 426: 233–271, 1989. All references herein are to the SRC Research Report 39 as revised Feb. 22, 1990.

[BAN91] Michael Burrows, Martín Abadi, and Roger Needham. The Scope of a Logic of Authentication. In Joan Feigenbaum and Michael Merritt, editors, *Distributed Computing and Cryptography*, volume 2 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 199–126. AMS and ACM, 1991.

[BGH$^+$92] Ray Bird, Inder Gopal, Amir Herzberg, Phil Janson, Shay Kutten, Refik Molva, and Moti Yung. Systematic Design of Two-Party Authentication Protocols. In Joan Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*. Springer Verlag, Berlin, 1992.

[Bie90] Pierre Bieber. A Logic of Communication in Hostile Environment. In *Proc. Computer Security Foundations Workshop III*. IEEE Computer Society Press, Los Alamitos, California, June 1990.

[CES86] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, April 1986.

[DvOW92] Whitfield Diffie, Paul C. van Oorschot, and Michael J. Wiener. Authentication and Authenticated Key Exchanges. *Designs Codes and Cryptography*, 2:107–125, 1992.

[FH88] Ronald Fagin and Joseph Y. Halpern. I'm OK if You're OK: On the Notion of Trusting Communication. *Journal of Philosophical Logic*, 17(4):329–354, November 1988.

[GNY90] Li Gong, Roger Needham, and Raphael Yahalom. Reasoning about Belief in Cryptographic Protocols. In *Proceedings of the 1990 IEEE Symposium on Security and Privacy*, pages 234–248. IEEE Computer Society Press, Los Alamitos, California, 1990.

[Gol92] Robert Goldblatt. *Logics of Time and Computation, 2^{nd} edition*, volume 7 of *CSLI Lecture Notes*. CSLI Publications, Stanford, 1992.

[HM90] Joseph Y. Halpern and Yoram Moses. Knowledge and Common Knowledge in a Distributed Environment. *JACM*, 37(3):549–587, July 1990.

[Mea91] C. Meadows. A System for the Specification and Analysis of Key Management Protocols. In *Proceedings of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 182–195. IEEE Computer Society Press, Los Alamitos, California, 1991.

[Mea92] C. Meadows. Applying Formal Methods to the Analysis of a Key Management Protocol. *Journal of Computer Security*, 1:5–53, 1992.

[SM93] Paul Syverson and Catherine Meadows. A Logical Language for Specifying Cryptographic Protocol Requirements. In *Proceedings of the 1993 IEEE Computer Society Symposium on Research in Security and Privacy*. IEEE Computer Society Press, Los Alamitos, California, 1993. Forthcoming.

[Sne92] Einar Snekkenes. Roles in Cryptographic Protocols. In *Proceedings of the 1992 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 105–119. IEEE Computer Society Press, Los Alamitos, California, 1992.

[Syv] Paul F. Syverson. On Key Distribution Protocols for Repeated Authentication. *Operating Systems Review*. Forthcoming.

[Syv91] Paul F. Syverson. The Use of Logic in the Analysis of Cryptographic Protocols. In *Proceedings of the 1992 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 156–170. IEEE Computer Society Press, Los Alamitos, California, 1991.

[Syv92] Paul F. Syverson. Knowledge Belief and Semantics in the Analysis of Cryptographic Protocols. *Journal of Computer Security*, 1(3):317–334, 1992.