Extending Cryptographic Logics of Belief to Key Agreement Protocols

(Extended Abstract)

Paul C. van Oorschot

Bell-Northern Research
P.O. Box 3511, Station C, Ottawa, Canada K2C 1Y7
paulv@bnr.ca

address until July 31 1994:
School of Computer Science, Carleton University, Ottawa, Canada K1S 5B6 (paulv@scs.carleton.ca)

Abstract. The authentication logic of Burrows, Abadi and Needham (BAN) provided an important step towards rigourous analysis of authentication protocols, and has motivated several subsequent refinements. We propose extensions to BAN-like logics which facilitate, for the first time, examination of public-key based authenticated key establishment protocols in which both parties contribute to the derived key (i.e. key agreement protocols). Attention is focussed on six distinct generic goals for authenticated key establishment protocols. The extended logic is used to analyze three Diffie-Hellman based key agreement protocols, facilitating direct comparison of these protocols with respect to formal goals reached and formal assumptions required.

1 Introduction

Authentication protocols serve a fundamental role in the cryptographic security of many systems, including the control of access to restricted areas, computer systems, and wireless telecommunications systems, and authentication in electronic banking transactions. The history of authentication protocols has highlighted the extreme difficultly of designing efficient authentication protocols which contain neither redundancies nor security flaws. The literature contains numerous examples of published protocols whose supposed correctness, as established by ad-hoc techniques and informal arguments, proved fleeting as subsequent examination revealed serious security weaknesses (e.g. see [3]). This has suggested the need for more rigourous methods to examine the correctness of authentication and associated key exchange protocols. To this end, Burrows, Abadi and Needham defined a logic of authentication (BAN) to allow formal modelling and exploration of beliefs in such protocols [3], [4]. Gaarder and Snekkenes (GS) extended the logic to allow further reasoning about public-key based protocols, and to capture the notion of "duration" related to timestamps; they then carried out a detailed analysis [11] of the public-key based X.509 two-way authentication protocol [5]. Related cryptographic logics of belief have been proposed to address recognized deficiencies [9] of BAN, including those of Gong, Needham and Yahalom (GNY) [10], and Abadi and Tuttle (AT) [1]. The X.509 analysis notwithstanding, much of the focus of research in this area has been on protocols involving on-line trusted servers and keys generated by a single party (symmetrically). Research to date has encompassed public key techniques for authentication and key transport, but not for key generation. More specifically, public-key based key establishment protocols in which both parties contribute to the established key — referred to as *key agreement* protocols — have not previously been analyzed by these methods. We propose extensions of BAN and BAN-like logics to facilitate more precise identification and examination of the goals and beliefs arising in authenticated key agreement protocols. We then illustrate the modified logic by examining three quite distinct such protocols based on Diffie-Hellman key exchange [6], including one which is identity-based.

The remainder of the paper is organized as follows. Section 2 (with Appendix A) reviews the essential features of the logic, and introduces new extensions including a refinement of the fundamental BAN construct "shares the good crypto key", a new primitive regarding key confirmation, and new postulates which facilitate for the first time reasoning about jointly established keys. Section 3 highlights six fundamental candidate formal goals for authenticated key establishment protocols. Section 4 gathers in one place the various formal assumptions required in our subsequent analyses. Section 5 applies the logic to the Station-to-Station (STS) protocol [7], Section 6 analyzes the Goss protocol [12], and Section 7 analyzes the Günther protocol [13]. Section 8 uses these results to compare the assumptions and goals of these protocols, and the aforementioned X.509 protocol [5]. Section 9 provides concluding remarks.

This work was originally framed entirely on that of BAN and GS. As this resulted in our work inheriting several of the known deficiencies in BAN [9], we have made selective use of more recent advancements, primarily from GNY and AT. Familiarity with GNY and AT is assumed. Where appropriate, we comment how the new extensions apply to these logics; further such examination remains to be done.

2 Authentication logic refinements and extensions

The BAN logic, with minor extensions by GS, is reviewed in Appendix A. In this section we propose new refinements to allow more precise reasoning for protocols involving jointly established keys. In what follows, A and B denote principals (or parties); U is used to denote a principal when we wish to specifically emphasize that its identity is unknown or uncorroborated. For clarity, we use the AT nomenclature "*said*" in place of the more verbose BAN "*once_said*". To denote that A has sent a formula X in the present epoch (i.e. has recently said X), we use the AT construct "A *says* X" (whereas in BAN, "A $\mid\equiv$ X" is used to denote both this and the fact that A believes X). This requires use of AT axiom A20 (*fresh*(X) $\wedge$ P *said* X $\supset$ P *says* X) in place of the BAN nonce-verification rule (R2 in Appendix A).

Version dated August 12, 1993. This paper will appear in the *Proceedings of the 1st ACM Conference on Communications and Computer Security*, November 3-5, 1993, Fairfax, Virginia.

We begin with a few new constructs and notation. In order to reason more precisely about cryptographic keys (hereafter called *keys*), the concept *shares the good key*,[1] denoted by the symbol A $\overset{K}{\leftrightarrow}$ B, requires refinement. This is necessary both to remove ambiguity, and to help avoid confusion about the meaning and (mis)use of the symbol (e.g. see [16]). For example, whether or not B actually knows K, A |≡ A $\overset{K}{\leftrightarrow}$ B is used in BAN to mean A believes that K is a good key for use with party B. Here the key is "good" in the sense that if and when it eventually becomes known to B, it will be safe to use for secure communications. For our purposes, more precision is required. One step in this direction is the AT construct "A *has* K", meaning that A actually possesses K.[2] It is important to note that possession is quite distinct from the notion of holding any beliefs about the quality or properties of K (e.g. K is a good key, K is shared with B). Without further information (e.g. whether K is also known by some unidentified or uncorroborated party B), such a key K which A *has* is called an *"unqualified*" key (from A's point of reference). Supplementing this refinement we now replace A $\overset{K}{\leftrightarrow}$ B with the following as appropriate:

A $\overset{K-}{\leftrightarrow}$ B — K is A's *unconfirmed secret* suitable for B. No one aside from A and B knows or could deduce K. This construct emphasizes, however, that while A knows K, the specific party B may or may not. A may consider K a "*qualified*" or "*secure*" key for use with B.

A $\overset{K+}{\leftrightarrow}$ B — K is A's *confirmed secret* suitable for B. A knows K, and has received evidence (*key confirmation*) from B indicating that B actually knows K. No other parties know or can deduce K.

Note that in these new constructs, A and B are not interchangeable. We reserve the term *authenticated key establishment* for mechanisms providing keys which are both secure and confirmed, in the above sense. Our motivation is alignment with the term *entity authentication*, which is not necessarily provided by mechanisms establishing (only) unconfirmed secrets. By this terminology, note that secure key establishment does not require entity authentication (in Diffie et al. [7], this is discussed in terms of *direct* authentication and *indirect* authentication). In what follows, the STS protocol is seen to provide authenticated key establishment; the Goss and Günther protocols provide unconfirmed secrets.

We also refine the symbols PK(K, A) and ∏(A) from BAN and GS. One reason is to distinguish the use of asymmetric key pairs for signatures, encryption, and key agreement, forcing explicit acknowledgment when one key pair is used for multiple purposes; this also precludes the incorrect assumption that signature key pairs can be used as encryption key pairs in all cryptosystems. A second reason is to separate the notion of *binding* a public key to a principal from the notion of the *goodness* of that key; we specifically associate "goodness" with the private key of a key pair. The symbols we use in place of these are:

PK_σ(A, K) — K is the *public signature verification key* associated with principal A.

PK^{-1}_σ(A) — A's *private signature key* K^{-1} *is good*. Here K^{-1} corresponds to the public key K in PK_σ(A, K). The key is "good" in that it is known only to A, cannot be derived by others, and does not result in a "weak" public key susceptible to specialized attacks.

PK_δ(A, K) — K is the *public key-agreement key* associated with principal A. When the specific value of the key is not of central focus or is evident by context, we write simply PK_δ(A).

PK^{-1}_δ(A) — A's *private key-agreement key* K^{-1} *is good*. Here K^{-1} corresponds to the public key K in PK_δ(A, K). The key is "good" in that it is known only to A, cannot be derived by others, and does not result in a "weak" public key (e.g. arising from a trivial exponential such as "1" in a Diffie-Hellman key exchange).

For asymmetric encryption keys, we suggest defining PK_ψ(A, K) and PK^{-1}_ψ(A) analogously; however, we do not require these symbols in the present work.[3] We also introduce a notational convenience, an *origin-mapping* construct, and a *knowledge-demonstration* construct:

⊥Y — This denotes the *key value associated with key symbol* Y. This allows one, for example, to use $\perp PK^{-1}_\delta$(A) to denote the value of the implied private key, in the absence of an explicit name (e.g. "K") for the associated public key.

$G(R_A)$ — $G(R_A)$ = {principals U: U *said* R_A}. This denotes the party U (or set of all parties U) which once conveyed or sent the nonce R_A. It facilitates reasoning about random numbers serving as challenges in challenge-response protocols (see Appendix A). This allows refinement of the time period "current epoch" (see Appendix A) to the more specific notion of "current run", to address "interleaving attacks" as discussed by Bird et al. [2].

confirm(K) — *Current knowledge* of K *has been demonstrated* (without compromising K). Note demonstration of knowledge of K differs from both actual and claimed possession of K. In particular, "A |≡ ∗*confirm*(K)"[4] differs subtly but significantly from "A |≡ U *has* K"; while the latter belief could arise even if U does not possess K,[5] the former requires "hard evidence".

The semantics of *confirm*(K) are best understood in light of the following new Confirmation Axioms:[6]

$$\text{C1.}\quad \textit{fresh}(X) \wedge \phi(\{X\}_K) \supset \textit{confirm}(K)$$

$$\text{C2.}\quad \textit{fresh}(X) \wedge \phi(MAC_K(X)) \supset \textit{confirm}(K)$$

$$\text{C3.}\quad \textit{fresh}(K) \wedge \phi(H(K)) \supset \textit{confirm}(K)$$

These specify that current knowledge of a key K can be demonstrated by: encrypting a fresh formula X with K; computing a message authentication code (MAC) over a fresh message, with key K;

[1] GNY uses the more generic semantics "K is a *suitable secret* for the two parties", which we find preferable.

[2] A similar GNY construct, "A *possesses* K", means that A either has, or is capable of computing, K.

[3] The subscript characters sigma and psi were chosen as memory-aids for the words *sig*nature and *ci*phering; delta was chosen for its association with a "changing" key, as key-agreement keys are often short-term (session) keys.

[4] Here "∗" is the GNY "not-originated-here" symbol, intended to formalize the implicit BAN assumption that parties can distinguish messages they generate from those generated by other principals.

[5] For example, if A granted U jurisdiction on claims of possession, and U lied; or if beliefs are based on messages sent — called "eager" beliefs in [14] — but not necessarily received (e.g. using GNY P1, P2, and GNY rationality rule).

[6] Here "ϕ" is the GNY "recognizability" construct, which formalizes the implicit BAN assumption of sufficient a priori knowledge, or redundancy, in encrypted messages to allow recognition of correct decryption keys.

or hashing a fresh key K using a one-way hash function H. Other similar axioms could be specified, but these suffice for our present purposes.

The original BAN postulates are listed in Appendix A. We now introduce three new postulates, based on the following model for public key agreement: each of two parties involved in a joint key agreement has a public and a private key-agreement key, and can derive the common (jointly established) key from their own private key and the other party's public key, using some agreement function *f*(private_info, public_info); here each parameter may actually be composite. One example of *f* is Diffie-Hellman key agreement [6]. We assume *f* results in a joint key K which can be deduced only with knowledge of the appropriate private information, and that knowledge of K does not compromise the secrecy of the other party's private information. (Recall that logics of belief generally assume soundness of all underlying cryptographic mechanisms.) The new postulates are:

R30. (Unqualified key-agreement rule):[1]

$$\frac{\text{A } \textit{has } PK^{-1}_{\delta}(\text{A}),\ \ \text{A } \textit{has } PK_{\delta}(\text{U})}{\text{A } \textit{has } \text{K}}$$

where $\text{K} = f(\,PK^{-1}_{\delta}(\text{A}), PK_{\delta}(\text{U})\,)$

By R30, A can compute a joint key from a private key-agreement key of her own and a public key-agreement key of some other party; this is basically the model for key agreement. A should treat this joint key as an unqualified key, as the binding between party U and its public key may be uncorroborated. R30 is a specific instance of GNY possession rule P2 which defines computability (A *has* X $\wedge$ A *has* Y $\supset$ A *has* F(X,Y)). Using P2 we also require GNY P1 (A *sees* X $\supset$ A *has* X).

R31. (Qualified key-agreement rule):

$$\frac{\text{A} \mid\equiv PK^{-1}_{\delta}(\text{A}),\ \ \text{A} \mid\equiv PK_{\delta}(\text{B}),\ \ \text{A} \mid\equiv PK^{-1}_{\delta}(\text{B})}{\text{A} \mid\equiv \text{A} \overset{\text{K-}}{\leftrightarrow} \text{B}}$$

where $\text{K} = f(PK^{-1}_{\delta}(\text{A}), PK_{\delta}(\text{B})\,)$

In R31, party A has another party's public key-agreement key, believes the binding of that key to the claimed identity, and believes the corresponding private key, in addition to her own, is good. This allows A to believe that the resulting key-agreement key is a qualified or secure (but unconfirmed) key.

R32. (Key confirmation rule):

$$\frac{\text{A} \mid\equiv \text{A} \overset{\text{K-}}{\leftrightarrow} \text{B},\ \ \text{A } \textit{sees } *\textit{confirm}(\text{K})}{\text{A} \mid\equiv \text{A} \overset{\text{K+}}{\leftrightarrow} \text{B}}$$

R32 allows A to upgrade her belief in the quality of key K — from a qualified to a confirmed key — upon observing evidence that a party other than herself knows the key. Note the definition of $\text{A} \overset{\text{K-}}{\leftrightarrow} \text{B}$ guarantees there is at most one other such party, namely B. In the original BAN syntax, the pre-condition "A *sees* $*$*confirm*(K)" might be stated as "*fresh*(X) $\wedge$ A *sees*$\{\text{X}\}_{\text{K from U}}$, where U≠A". R32 is a message interpretation rule (cf. BAN R1, Appendix A). It has much in common with GNY rule I1 [10], although the GNY definition of "B *possesses* K" results in I1 not distinguishing between B knowing/having K and being capable of computing K, nor does it imply B has demonstrated knowledge of K.

[1]To be technically precise, we should use $\perp PK^{-1}_{\delta}(\text{A})$ and $\perp PK_{\delta}(\text{U})$ in place of $PK^{-1}_{\delta}(\text{A})$ and $PK_{\delta}(\text{U})$ in the statement of R30, and as arguments to the key agreement function *f*, but we write simply the latter for appearance; the meaning should be clear by context. A similar comment applies to the arguments of *f* in R31.

3 Generic formal goals

The apparently obvious goal of an authentication protocol is the provision of some degree of assurance of the identity of another party. In authenticated key establishment, an intended outcome is the creation and/or distribution of a (possibly new) secret key. While these goals may appear obvious, as stated they are quite imprecise, and subtle differences exist among protocols regarding the exact properties established. Failure to understand the precise meaning of specific goals has lead to misunderstandings about the differences between various protocols. This motivates us to explicitly identify six distinct candidate positions which may or may not be intended as formal goals in a specific authenticated key establishment protocol. We state these as beliefs of party A, with party B the other intended party in the protocol.

(G1)	*Far-end operative*:	A $\mid\equiv$ B *says* Y
(G2)	*Targeted entity authentication*:[2]	A $\mid\equiv$ B *says* (Y, $R(G(\text{R}_\text{A}), \text{Y})$)
(G3)	*Secure key establishment:*	A $\mid\equiv \text{A} \overset{\text{K-}}{\leftrightarrow} \text{B}$
(G4)	*Key confirmation:*	A $\mid\equiv \text{A} \overset{\text{K+}}{\leftrightarrow} \text{B}$
(G5)	*Key freshness:*	A $\mid\equiv$ *fresh*(K)
(G6)	*Mutual belief in shared secret*:	A $\mid\equiv$ (B $\mid\equiv$ B $\overset{\text{K-}}{\leftrightarrow}$ A)

(G1) states that A believes B recently conveyed a message Y. It implies that B is currently operational (or "alive"), i.e. has taken action subsequent to the start of the protocol. Inherent is the fact that the identity of B has been corroborated, but it is unclear who B intended to convey the message to. Note "aliveness" also follows from (G2) and (G4), but not (G3).

(G2) states A believes a message Y was recently conveyed by B in response to the specific challenge R_A (R_A here is a "random number" — see Appendix A). It provides authentication of B to A in the sense that the response is from a corroborated operational entity, and is targeted in response to a (preferably fresh) challenge from A. That B's formula is specifically in response to A's challenge ties B's reply to the protocol run A is executing. Note while entity authentication requires parties be operative, key establishment protocols do not, as not all provide entity authentication; indeed, store-and-forward environments do not support on-line entity authentication.

(G3) states that A believes that a key K is shared with no party other than possibly party B. K is suitable for use by A with B if and when B eventually acquires it. (G3) does not imply that B participated in any manner in the protocol, nor even possesses K.

(G4) states A believes a key K is shared with party B alone, and that B has provided evidence of knowledge of the key to A. It implies both that the quality of the key is good, and confirmation that the far end has knowledge of K; aliveness and corroboration of identity of the far end party are inherent.

(G5) states A believes a key K is fresh. It addresses the possibility that a key might be reused or replayed.

[2]We hesitate to call this "entity authentication", due to the absence of a universal definition; our accompanying formal description clarifies our intended meaning. This particular expression of entity authentication is specifically based on challenge-response. For protocols that are not based on challenge-response (e.g. timestamps or sequence numbers), "$G(\text{R}_\text{A})$" might be replaced simply by "A" in this goal, but this changes the implications of the goal significantly.

(G6) states that A believes that party B believes K is an unconfirmed secret suitable for use with A. Note B's beliefs are beyond the control of A, and the beliefs of B of true importance to A are those which concern A directly. Thus of greater import to A here than B's (presumed) belief that K is secure is A's confidence that B has correctly identified her as the party with which B shares the trusted key. Competence on the part of B is assumed here by A.

Among other goals that might be stated, but upon which we do not dwell, include:

(G7) *Key possession:* A *has* K

(G8) *Belief in far-end possession*: A $\mid\equiv$ B *has* K

Since (G3) is a minimum goal for key establishment, and it inherently implies (G7), we view the latter as somewhat redundant. We feel that (G8), although of possible theoretical interest, is of questionable use in practice, with "physical world" evidence as given by (G4) being preferable. (See §6 of [10], §3.2 of [1], and §3.3 of [9] for related discussions.)

We note that (G4), i.e. $A \mid\equiv A \overset{K+}{\leftrightarrow} B$, is independent of B's beliefs about K, and is thus distinct from (G6). Previous BAN-like logics have lacked a suitable construct to express the concept of key confirmation. Also, from the definition of *confirm*(K), it should be clear that $A \mid\equiv A \overset{K+}{\leftrightarrow} B$ is *not* equivalent to the conjunction of (G3) and (G7), i.e. $(A \mid\equiv A \overset{K-}{\leftrightarrow} B) \wedge (A \mid\equiv B \textit{ has } K)$.

4 Generic formal assumptions

In this section we collect for reference candidate formal assumptions which the protocols subsequently analyzed require. In a typical BAN-type analysis, preliminary formal assumptions that appear necessary or "obvious" are recorded, and additional assumptions become apparent as necessary pre-conditions to use of logic postulates required in formal proofs. (Assumptions which appear intuitively necessary at the outset, but are not found to be required anywhere in the formal proofs, should be carefully re-examined.) Despite the latter, most analyses are presented "top-down", with the appearance that all assumptions are known *a priori*.[1] The assumptions below are stated as typical candidate requirements on a party A, with B the other intended participant.[2]

A believes she has[3] a valid copy of the public signature key (K_T) of the *trusted authority* T. A also believes that the corresponding private signature key is "good":

$$A \mid\equiv PK_\sigma(T, K_T) \qquad \text{(A1)}$$

$$A \mid\equiv PK^{-1}{}_\sigma(T) \qquad \text{(A2)}$$

A believes that T has jurisdiction over the binding of a public signature key (K_B) with a specific party (note T is not given jurisdiction over the quality of the corresponding private key). A similar assumption concerns a pre-certified public key-agreement key ($\overline{R}_B$):

$$A \mid\equiv T \textit{ controls } PK_\sigma(B, K_B) \qquad \text{(A3)}$$

$$A \mid\equiv T \textit{ controls } PK_\delta(B, \overline{R}_B) \qquad \text{(A4)}$$

A believes any private key-agreement key she herself uses is good:

$$A \mid\equiv PK^{-1}{}_\delta(A) \qquad \text{(A5)}$$

While it is necessary for A to guard her own private signature key so that the beliefs of others that her private key is "good" are valid, this is not necessary for A to establish her own beliefs; however (A5) is. A also believes all principals (e.g. B) have the competence to acquire "good" private keys for themselves, and to safeguard such keys; this is consistent with the basic assumption that all principals act in their own best interest (including to guard jointly derived secrets), and does not preclude the existence of attackers:

$$A \mid\equiv PK^{-1}{}_\sigma(B) \qquad \text{(A6)}$$

$$A \mid\equiv PK^{-1}{}_\delta(B) \qquad \text{(A7)}$$

A grants any principal B jurisdiction over his own public key-agreement key R_B (cf. (A4)):

$$A \mid\equiv B \textit{ controls } PK_\delta(B, R_B) \qquad \text{(A8)}$$

Note this does not grant to B jurisdiction over claims regarding the corresponding private key; corroborating evidence is required to back such claims. Thus while (A8) allows the possibility that B may claim another party's public key as his own (indeed, this is a practical possibility which the logic cannot simply "rule out"), note a dishonest such principal is unable to compute the associated secret key and provide key confirmation.

A believes a random number (R_A) she generates herself is *fresh*, i.e. was not used in a previous protocol run (note this does not require that A believe in the freshness of nonces created by other parties):

$$A \mid\equiv \textit{fresh}(R_A) \qquad \text{(A9)}$$

A final assumption is related to the (continuing) validity of public keys. Certificates are typically used to make one party's public key(s) available to other parties. A *certificate* is a block of information containing a party's *credentials* (the subject party's public key(s), identifying information such as distinguishing name, and possibly other information) together with the signature of a *trusted (certification) authority* T over the credentials. The validity of a certificated public key is verified by using T's public signature key to verify the signature on the certificate. The assumption we state is as follows:

$$\frac{A \mid\equiv (T \textit{ said } PK(B, K_B))}{A \mid\equiv (T \mid\equiv PK(B, K_B))} \qquad \text{(A10)}$$

Note this is *not* a universal inference rule; it is a novel usage of an inference rule *as an assumption*. The intent is to use this rule in place of the informal assumption that some procedure is available to verify that statements of T from the past, which bind a public key to a distinguishing name, hold true in the current run. The objective is that this assumption generically handle the large number of ways in which the validity of a public key may be controlled in practice (e.g. validity periods in certificates, revocation of certificates, etc.). Many alternatives to (A10) as given above exist. In a particular application, one might assume that a public key, once valid, is valid for all time; i.e. assume *a priori* that $A \mid\equiv T \mid\equiv PK_\sigma(B, K_B)$, as essentially done in the X.509 analysis of Burrows et al. [3]. A second approach is available if the concrete protocol is such that the validity period is constrained by a timestamp in the certificate, as in GS; this then requires further assumptions regarding timestamps and timestamp verification. A third is to assume that a public key in a certificate is valid as long as the signature key of T used to sign the certificate remains valid; this requires re-verification of the certificate signature each time the public key within

[1]Mao and Boyd have recently suggested formalizing the "bottom-up" approach to proofs, to systematically *derive* a minimum set of necessary pre-conditions, starting from a fixed set of goals [15].

[2]To follow GNY strictly, we would add further assumptions about "recognizability" of encrypted and signed values.

[3]Again, following GNY strictly, we would record additional assumptions regarding the initial possessions of A, and track subsequent acquisitions in annotated analysis. For example, (A1) implies both possession of a key and belief that the binding of it to T is good. For brevity, we typically treat possessions informally, as in BAN.

it is used, which could be handled formally by the GNY "recognizability" construct applied to signatures. The generic (A10) above conveniently handles these and other possibilities simultaneously.

5 Key exchange protocol #1: STS protocol

We first review the authenticated key agreement protocol of Diffie et al. called the "Station-to-Station" (STS) protocol [7]. A publicly known appropriate prime p and primitive element α in GF(p) are fixed for use in Diffie-Hellman key exchange. Parties A and B use a common signature scheme. $s_U\{\bullet\}$ denotes the signature on the specified argument using the private signature key of party U. $E_K(\bullet)$ denotes the encryption of the specified argument using algorithm E under key K. Public key certificates are used to make the public signature keys of A and B available to each other. In a one-time process prior to the exchange between A and B, each party must present to T their identity and public key (e.g. ID_A, K_A), have T verify their true identity by some (typically non-cryptographic) means, and then obtain from T their own certificate. The protocol analyzed is as follows.

1. A generates a random positive integer x, computes $R_A = \alpha^x$ and sends R_A to a second party.
2. Upon receiving R_A, B generates a random positive integer y, computes $R_B = \alpha^y$ and $K = (R_A)^y$.
3. B computes the authentication signature $s_B\{R_B, R_A\}$ and sends A the encrypted signature $Token_{BA} = E_K(s_B\{R_B, R_A\})$ along with R_B and his certificate $Cert_B$. Here "," denotes concatenation.
4. A receives these quantities, and from R_B computes $K = (R_B)^x$.
5. A verifies the validity of B's certificate $Cert_B$ by verifying the signature thereon using the public signature key of the authority. If the certificate is valid, A extracts B's public signature key, K_B, from $Cert_B$.
6. A verifies the authentication signature of B by decrypting $Token_{BA}$, and using K_B to check the signature on the decrypted token is valid for the known, ordered pair R_B, R_A.
7. A computes $s_A\{R_A, R_B\}$ and sends to B her certificate $Cert_A$ and $Token_{AB} = E_K(s_A\{R_A, R_B\})$.
8. Analogously, B checks $Cert_A$. If valid, B extracts A's public signature key, K_A and proceeds.
9. Analogously, B verifies the authentication signature of A by decrypting $Token_{AB}$, and checking the signature on it using K_A and knowledge of the expected pair of data R_A, R_B.

The protocol is successful from each party's point of view if signature verification succeeds on both the received certificate and authentication signature. In this case, the protocol provides assurance that a shared secret has been jointly established with the party identified in the received certificate.

TABLE 1 provides a summary of both the messages exchanged, and the actions taken, by each of the parties in this protocol.

5.1 Formal analysis of STS protocol

The protocol is first idealized into a form suitable for logic manipulation. With $K = \alpha^{xy}$ as above, define

$$R_A = \alpha^x, \qquad R_B = \alpha^y \qquad \text{and} \tag{1}$$

$$M_B = (T_B, R(G(R_A), T_B), PK_\delta(B, R_B)) \text{ with } T_B = (R_B, R_A) \tag{2}$$

$$M_A = (T_A, R(G(R_B), T_A), PK_\delta(A, R_A)) \text{ with } T_A = (R_A, R_B) \tag{3}$$

A's certificate is idealized as $\{PK_\sigma(A, K_A)\}_{sT}$. Note this idealization contains the signature over, but not the actual data values A, K_A. These latter cleartext data values, omitted in the BAN idealization ("as they do not contribute to the beliefs held, and do not enter into the analysis"), are nonetheless implicitly assumed available to the recipient for operational reasons; they would be included explicitly in a GNY idealization. Either approach suffices for our purposes, and indeed we use both as convenient. The idealized STS protocol is:

$$A \rightarrow B\colon\ R_A \tag{M1}$$

$$A \leftarrow B\quad R_B,\ (B, K_B, \{PK_\sigma(B, K_B)\}_{sT}),\ \{\{M_B\}_{sB}\}_K \tag{M2}$$

$$A \rightarrow B\colon\ (A, K_A, \{PK_\sigma(A, K_A)\}_{sT}),\ \{\{M_A\}_{sA}\}_K \tag{M3}$$

The idealization conveys some beliefs implicit, but not directly represented, in the actual protocol. For example, the binding of public key to name — $PK_\delta(B, R_B)$ — in M_B of (M2) is not explicit in the actual protocol, nor is the intended recipient of T_B in (M2), which is the party who challenged with R_A, i.e. $G(R_A)$. However, B's signature on T_B in the actual protocol implicitly conveys this information. Note that only the cryptographically protected messages, i.e. steps (M2) and (M3), will contribute directly to the logical beliefs that result.

The formal assumptions from Section 4 required of party A in the STS protocol are listed in TABLE 4 in Section 8. Analogous assumptions are required of B. Regarding the security of underlying algorithms in the STS protocol, use of rule R31 relies on an assumption regarding the particular function f used. The assumption here is the standard Diffie-Hellman assumption: given values R_A and R_B which are exponentials based on secret keys x and y, respectively, it is computationally infeasible to compute the corresponding Diffie-Hellman key K without knowledge of either x or y.

We now focus on the formal goals related to the final position of A. While the intended goals of the STS protocol were not stated [7] in the syntax of BAN-like logics, the goals from Section 3 it attains include: (G1), (G2), (G3), (G4), (G5) and (G6). We view the last three as the major end goals; these encompass the first three. We now outline proofs that these goals are actually attained.

Lemma 1 The STS protocol establishes that the far-end party is operative, i.e. achieves goal (G1).

Proof: Upon A's reception of (M2), R10 provides:

$$A \textit{ sees } R_B \tag{4}$$

TABLE 1 STS protocol (concrete)

A		B
$Cert_A = (K_A, ID_A, s_T\{K_A, ID_A\})$		$Cert_B = (K_B, ID_B, s_T\{K_B, ID_B\})$
generate x, $R_A = \alpha^x$	$\rightarrow R_A$	generate y, $R_B = \alpha^y$; $K = (R_A)^y$
$K = (R_B)^x$; verify $Cert_B$, $Token_{BA}$	R_B, $Cert_B$, $Token_{BA}$ $\leftarrow$	$Token_{BA} = E_K(s_B\{R_B, R_A\})$
$Token_{AB} = E_K(s_A\{R_A, R_B\})$	$\rightarrow Cert_A$, $Token_{AB}$	verify $Cert_A$ and $Token_{AB}$

$$\text{A } \textit{sees } \{PK_\sigma(\text{B}, \text{K}_\text{B})\}_{\text{sT}} \tag{5}$$

$$\text{A } \textit{sees } \{\{\text{M}_\text{B}\}_{\text{sB}}\}_\text{K} \tag{6}$$

By A's belief (and possession) of her private key-agreement key (A5), and (4) along with GNY rule P1 as noted earlier (*sees* implies *has*), R30 provides

$$\text{A } \textit{has } \text{K} \tag{7}$$

where $\text{K} = f(\perp PK^{-1}_\delta(\text{A}), \perp PK_\delta(\text{U}))$. K is an unqualified key, potentially shared with an uncorroborated party U. (7) and (13) with R22 yields

$$\text{A } \textit{sees } \{\text{M}_\text{B}\}_{\text{sB}}. \tag{8}$$

Assumption (A1), (A2) and (5) with R13 yields A $\models$ T *said* $PK_\sigma(\text{B}, \text{K}_\text{B})$, which (A10) strengthens to

$$\text{A} \models \text{T} \models PK_\sigma(\text{B}, \text{K}_\text{B}) \tag{9}$$

Then (9) and (A3) with R3 yields A $\models PK_\sigma(\text{B}, \text{K}_\text{B})$. This, (8) and (A6), with R13 provides

$$\text{A} \models \text{B } \textit{said } \text{M}_\text{B} \tag{10}$$

By assumption (A9), A $\models$ *fresh*(x). Exponentiation of the primitive element α by x induces a bijection, suggesting freshness propagation from x to $\text{R}_\text{A} = \alpha^x$ based on R12 (cf. GNY rule F1). This provides

$$\text{A} \models \textit{fresh}(\text{R}_\text{A}) \tag{11}$$

Then R12 further yields A $\models$ *fresh*(T_B), and freshness propagation again (note from (13) T_B and M_B are cryptographically sealed) allows

$$\text{A} \models \textit{fresh}(\text{M}_\text{B}) \tag{12}$$

Now (10) and (12) with AT axiom A20 (cf. BAN R2) yields

$$\text{A} \models \text{B } \textit{says } \text{M}_\text{B} \tag{13}$$

Thus A believes that B recently said M_B. This is (G1) with $\text{Y}=\text{M}_\text{B}$ as defined in (2).

❑

Lemma 2 The STS protocol provides targeted entity authentication, i.e. achieves goal (G2).

Proof: From (13) of Lemma 1 and R7, A $\models$ B *says* (T_B, $R(G(\text{R}_\text{A}), \text{T}_\text{B})$). This is precisely (G2) with $\text{Y}=\text{T}_\text{B}$ as given in (2); here $\text{R}_\text{A} = \alpha^x$. Thus upon successful completion, A believes that B conveyed T_B in the current epoch, as an intended response to the specific challenge R_A.

❑

Provided A does not intentionally re-use nonces, and generates a nonce x (and $\text{R}_\text{A} = \alpha^x$) in the current epoch using an appropriate random number generator (producing unpredictable numbers from a sufficiently large space, with vanishing probability of repetition), the nonce will be a duplicate of a previous nonce with vanishing probability. Then whereas G is a one-to-many mapping on an unrestricted domain, A can conclude that with vanishing probability of error, $G(\text{R}_\text{A})$ is the singleton set {A}. In this case Lemma 2 allows A to conclude she was the intended recipient of B's token, i.e. A $\models$ B *says* $R(\text{A}, \text{T}_\text{B})$. Both Lemma 1 and Lemma 2 rely directly on (A9).

Lemma 3 The STS protocol provides secure key establishment, i.e. achieves goal (G3).

Proof: From Lemma 1 and R6 it follows that A $\models$ B *says* $PK_\delta(\text{B}, \text{R}_\text{B})$. Using this and B's jurisdiction over his public key (A8), R3 yields A $\models PK_\delta(\text{B}, \text{R}_\text{B})$. By A's belief in private key-agreement keys (A5) and (A7), R31 then yields A $\models \text{A} \overset{\text{K-}}{\leftrightarrow} \text{B}$. That is, A believes K is shared with no party other than possibly B. Implicitly, A also now possesses this key.

❑

Lemma 4 The STS protocol provides key confirmation, i.e. achieves goal (G4).

Proof: [*Outline only*] In the BAN logic, the proof is short: from Lemma 3 and (13), R32 (modified for BAN as discussed in Section 2) yields the result directly. However, this is unsatisfying due to the recognized limitation of the BAN construct $\{\text{X}\}_{\text{K from U}}$ (see below). Consequently, we provide the following alternate proof outline using GNY constructs. We require one additional formal assumption:[1] A $\models \phi(\text{M}_\text{B})$, from which GNY recognition rules R2 and R3 provide A $\models \phi(\{\{\text{M}_\text{B}\}_{\text{sB}}\}_\text{K})$. From (12), freshness propagation (GNY F1) allows the conclusion A $\models$ *fresh*($\{\text{M}_\text{B}\}_{\text{sB}}$). Confirmation Axiom C1 (Section 2) and these two beliefs yield: A *sees confirm*(K). As A creates no message of the specific form $\{\{\text{M}_\text{B}\}_{\text{sB}}\}$ in the protocol, M_B would be marked with a "not-originated-here" symbol — $*\text{M}_\text{B}$ — following GNY protocol parsing. The confirmation belief is then: A *sees* ∗*confirm*(K). This, with Lemma 3 and R32, allows the conclusion A $\models \text{A} \overset{\text{K+}}{\leftrightarrow} \text{B}$. That is, upon successful completion, A believes K is shared with B alone, and that B has provided to A evidence of his knowledge of this key.

❑

Lemma 5 The STS protocol provides key freshness, i.e. achieves goal (G5), provided B does not choose $y \equiv 0 \pmod{p-1}$.

Proof: As in (11), A believes that α^x is a nonce and a random element of the field. For non-zero y, the entropy of $\text{K} = (\alpha^x)^y$ is then large (even in the worst case of "smooth" primes p; see [17]). Therefore freshness propagation (by R12 or GNY F1) over this exponentiation provides freshness of the key K.

❑

Note A's belief in key freshness is "pure" in the sense that it is based only on factors within her own control; it requires (A9), but no trust or honesty in other parties. This differs, for example, from a belief that a key from a trusted server is fresh simply because the key is integrated with a nonce.

We finally consider goal (G6). Upon receipt of (M2), what may A deduce about B's beliefs? From Lemma 5 it follows that A may believe B possesses K, and although we do not provide details, one may derive A $\models$ (B $\models \text{B} \overset{\text{K-}}{\leftrightarrow} \text{U}$). However, as A does not identify herself until (M3), it is clear B cannot yet know U=A; indeed, B is anticipating $\text{U}=G(\text{R}_\text{A})$, but cannot verify this before receiving (M3). However, as noted following Lemma 2, A may deduce $G(\text{R}_\text{A})$=A, and may thus arrive at A $\models$ (B $\models \text{B} \overset{\text{K-}}{\leftrightarrow} \text{A}$) as an "eager belief" (using the terminology of [14]). This belief is "eager" in that it anticipates B's receipt of the final message, but the belief is not reinforced within the protocol as A receives no further messages. We state this eager belief without further proof.

Lemma 6 The STS protocol provides mutual belief in a shared keying relationship, i.e. achieves goal (G6); this is however an "eager" belief on A's part.

❑

Now consider the beliefs of party B after successfully completing the STS protocol. B is able to arrive at beliefs analogous to those of party A given above. The proofs are analogous, except that in

[1] This recognizability assumption is implicit in the BAN proof, and follows from the action of successful verification of B's signature in (13). This is the main reason we find our BAN-logic proof unsatisfying.

B's case, goal (G6) may be attained without qualification. This results from B having the advantage of being recipient of the final message. Note the mere presence of message (M3) implies A has successfully completed her end of the protocol, and arrived at her final beliefs. Confirmation of A's successful completion could be explicitly modelled in the idealized protocol, e.g. by incorporating appropriate beliefs into the signed token M_A in idealized message (M3).[1] Granting additional assumptions giving A jurisdiction over her own final beliefs, these beliefs would then be derivable as mutual beliefs of B. For example, Lemma 3 would lead to B |≡ (A |≡ A $\overset{K}{\leftrightarrow}$ B), i.e. (G6) for B.

6 Key exchange protocol #2: Goss protocol

The key agreement protocol of Goss [12] results in the establishment of a shared secret key; two Diffie-Hellman exponentiations are used, combining fixed and (per-run) variant parameters, allowing the creation of a unique key for each protocol run while re-using certified public key-agreement keys. A publicly known appropriate prime p and primitive element α in GF(p) are fixed. The parties A and B and the trusted authority T use a common signature scheme in association with certificates; $s_U\{\bullet\}$ denotes the signature of party U as before. In a preliminary, one-time process, A selects a secret random number $\bar{x}$, computes $\bar{R}_A = \alpha^{\bar{x}}$, and gives this to T; T verifies A's identity and returns a certificate $Cert_A$ consisting of $\bar{R}_A$, a distinguishing identifier ID_A for A, and T's signature over their concatenation. $\bar{R}_A$ serves as A's fixed public key-agreement key, which can now be made available to others by certificate. Similarly, B obtains a secret number $\bar{y}$, computes $\bar{R}_B = \alpha^{\bar{y}}$, and obtains $Cert_B$. The protocol between A and B then consists of a single message in each direction, as outlined below and summarized in TABLE 2:

1. A generates a random integer x>0, computes $R_A = \alpha^x$ and sends R_A to B along with certificate $Cert_A$.
2. B generates a random integer y>0, computes $R_B = \alpha^y$ and sends R_B to A along with certificate $Cert_B$.
3. A and B establish the authenticity of each other's certificates by verifying the signature of T thereon using T's known public key, and establish a common key K by respectively computing $K = (\bar{R}_B)^x \oplus (R_B)^{\bar{x}}$ and $K = (\bar{R}_A)^y \oplus (R_A)^{\bar{y}}$.

TABLE 2 Goss protocol (concrete)

A				**B**
generate $\bar{x}$, $\bar{R}_A = \alpha^{\bar{x}}$				generate $\bar{y}$, $\bar{R}_B = \alpha^{\bar{y}}$
$Cert_A = (\bar{R}_A, ID_A, s_T\{\bar{R}_A, ID_A\})$				$Cert_B = (\bar{R}_B, ID_B, s_T\{\bar{R}_B, ID_B\})$
generate x, $R_A = \alpha^x$	$\rightarrow$	$Cert_A$, R_A		generate y, $R_B = \alpha^y$
verify $Cert_B$; $K = (\bar{R}_B)^x \oplus (R_B)^{\bar{x}}$		$Cert_B$, R_B	$\leftarrow$	verify $Cert_A$; $K = (\bar{R}_A)^y \oplus (R_A)^{\bar{y}}$

6.1 Formal analysis of Goss protocol

The protocol must first be idealized. A's certificate is idealized as $\{PK_\delta(A, \bar{R}_A)\}_{sT}$. Note here the public key bound is a key-agreement key rather than a signature key. The idealized Goss protocol is:

$$A \rightarrow B:\ (A, \bar{R}_A, \{PK_\delta(A, \bar{R}_A)\}_{sT}),\ R_A \qquad \text{(M4)}$$

$$A \leftarrow B:\ (B, \bar{R}_B, \{PK_\delta(B, \bar{R}_B)\}_{sT}),\ R_B \qquad \text{(M5)}$$

The idealization from the concrete protocol to the above form is straightforward. As in Section 5.1, only cryptographically protected information contributes directly to the establishment of logical beliefs, and thus the cleartexts R_A and R_B could be omitted from the idealization.

We now turn to formal assumptions. The assumptions of Section 4 required by party A in the Goss protocol are listed in TABLE 4 in Section 8; analogous assumptions are required of B. Regarding the security of underlying algorithms in the Goss protocol, use of R31 requires the following assumptions about the function f: given exponentials R_A, R_B, $\bar{R}_A$, and $\bar{R}_B$, it is computationally infeasible to compute the key K without knowledge of (x, $\bar{x}$) or (y, $\bar{y}$); and knowledge of one of these pairs, together with the first four values, does not allow one to recover the other pair.

We now consider the protocol goals. Informally, the Goss protocol is a technique "in which two users establish a common session key by exchanging information over an insecure communication channel, and in which each user can authenticate the identity of the other" [12]. The formal goal which can be proven reachable by party A[2] upon protocol completion is (G3), i.e. A |≡ A $\overset{K-}{\leftrightarrow}$ B. Corroboration that B actually knows the key K, i.e. (G4), while not part of the basic protocol, could be achieved by a subsequent message making use of K. We also note key freshness, i.e. (G5), as a reachable goal.

Lemma 7 The Goss protocol provides secure key establishment, i.e. achieves goal (G3).

Proof: Upon receiving (M5), A *sees* $\{PK_\delta(B, \bar{R}_B)\}_{sT}$. Using (A1), (A2) and this in R13 provides A |≡ T *said* $PK_\delta(B, \bar{R}_B)$, which (A10) strengthens to A |≡ T |≡ $PK_\delta(B, \bar{R}_B)$. This and (A4) with R3 yields A |≡ $PK_\delta(B, \bar{R}_B)$. From here, using A's belief in the goodness of the private key-agreement keys of both A and B — (A5) and (A7) — in R31 provides A |≡ A $\overset{K-}{\leftrightarrow}$ B, i.e. A believes K is shared with no party other than possibly B. Here the fixed certified key $\bar{R}_B = \alpha^{\bar{y}}$ plays the role of B's public key-agreement key in R31, A's fixed secret key $\bar{x}$ plays the role of A's private key-agreement key, and the uncertified time-variant keys (x and R_B) are secondary private and public parameters, respectively, for the key agreement function f.

❑

Lemma 8 The Goss protocol provides key freshness, i.e. achieves goal (G5), provided B does not choose $y \equiv 0 \pmod{p-1}$ or $\bar{y} \equiv 0 \pmod{p-1}$.

Proof: Similar to proof of Lemma 5.

❑

Note freshness assumption (A9) is used by Lemma 8 but not by Lemma 7.

7 Key exchange protocol #3: Günther protocol

The authenticated key agreement protocol of Günther [13] is an identity-based key establishment protocol, employing the idea of identity-based schemes for signatures/authentication, Diffie-Hell-

[1] This is essentially equivalent to "message extension" in GNY.

[2] The protocol being essentially identical from either party's perspective, we consider only the goal of the initiator.

man key agreement [6], and ElGamal signatures [8]. An appropriate public prime p and primitive element α in GF(p) are fixed. The trusted authority T chooses an integer v as its secret key, $1 \le v \le p-1$, and makes (K_T =) $u = \alpha^v$ public. In a preliminary, one-time process, it also assigns to each party a unique identifier D, and for each D chooses a random integer k_D with $\gcd(k_D, p-1) = 1$ and computes $r_D = \alpha^{k_D}$; then with h() a suitable hash function, solves the following equation for s_D (re-choosing k_D if s_D=0):

$$h(D) \equiv v \cdot r_D + k_D \cdot s_D \pmod{p-1} \qquad (14)$$

The pair (r_D,s_D) is an ElGamal signature on identifier D, which T gives to party D for safekeeping. r_D is publicly available; s_D is D's secret key allowing subsequent secure key establishment as outlined below. For use in what follows, note (r_D,s_D) satisfies the equation

$$\alpha^{h(D)} \equiv u^{r_D} \cdot r_D^{\,s_D} \pmod{p}, \quad \text{and hence} \qquad (15)$$

$$r_D^{\,s_D} \equiv \alpha^{h(D)} \cdot u^{-r_D} \pmod{p} \qquad (16)$$

Note $r_D^{\,s_D}$ can thus be computed entirely from publicly available information.[1] The protocol between A and B, which take the place of "D" above, consists of steps as follows:

1. A sends to B the pair (A, r_A); similarly B sends to A the pair (B, r_B).
2. A generates a random positive integer x, and sends to B the quantity $(r_B)^x$; similarly B generates a random positive integer y, and sends to A the quantity $(r_A)^y$.
3. A computes $\overline{R}_B$, where $\overline{R}_B = \alpha^{h(B)} \cdot u^{-r_B}$ (= $r_B^{\,s_B}$ from (16)), and $K = (r_A^{\,y})^{s_A} \cdot \overline{R}_B^{\,x}$; similarly B computes $\overline{R}_A$, where $\overline{R}_A = \alpha^{h(A)} \cdot u^{-r_A}$ (= $r_A^{\,s_A}$), and $K = (r_B^{\,x})^{s_B} \cdot \overline{R}_A^{\,y}$.

Both parties then share the key $K = (r_A^{\,s_A})^y\, (r_B^{\,s_B})^x$. TABLE 3 summarizes.

housing the key $\overline{R}_A$ is idealized as $\{PK_\delta(A, \overline{R}_A)\}_{sT}$. The idealization is as follows (compare to Goss, from which this idealization was motivated):

$A \rightarrow B$: A, r_A (M6)

$A \leftarrow B$: B, r_B, R_B (M7)

$A \rightarrow B$: R_A (M8)

A recovers $\{PK_\delta(B, \overline{R}_B)\}_{sT}$; B recovers $\{PK_\delta(A, \overline{R}_A)\}_{sT}$ (M9)

$\overline{R}_A$ and $\overline{R}_B$ are not transmitted by A and B, respectively, to the other, but rather are computed from unprotected data transmitted during the protocol and publicly available information (see (16)). As it is assumed only T can produce a pair (r_A, s_A) satisfying (16) for A, $\overline{R}_A$ is viewed as a public key pre-certified (by construction) by T; this pre-certification is idealized as a certificate. B is able to reconstruct $\overline{R}_A$; its authenticity is implicit in the expected equality of the resulting keys K computed by A and B. Here the idealization is no longer restricted to data from message exchanges; idealization is extended to apply to the data computed by parties, i.e. resulting from parties' *actions* within the protocol.[3] The time-invariant parameters (A, r_A) and (B, r_B) transmitted in the concrete protocol are not protected cryptographically, but their integrity is implicitly verifiable by the identity-based nature of the scheme. The same is true of the exchanged cleartexts R_A and R_B — in fact, the protocol contains no messages which are explicitly cryptographically protected.

We next consider formal assumptions of the Günther protocol. Those of Section 4 required by party A in the protocol are listed in TABLE 4 in Section 8; analogous assumptions are required of B. Here, since s_A is a secret generated by T, (A5) also implies the assumption that T is trusted to generate and securely transfer this secret to A without disclosing it to any other party; the same holds true for (A7). (A10) here applies to the validity of the secret signature pair (r, s) computed in the past by T, and used in the present as the certified public key r^s. Regarding the security of underlying algorithms in the Günther protocol, use of R31 in proofs of beliefs for this protocol requires the following assumptions about the key agreement function f: given values R_A, R_B, $\overline{R}_A$, and $\overline{R}_B$, as defined in Section 7.1, it is computationally infeasible to compute the key K without knowledge of (x, s_A) or (y, s_B); knowledge of one of these pairs, together with the first four values, does not allow one to recover the other pair; and a solution (r, s) to (14) requires knowledge of v. Some redundancy is typically embedded in D of (14) to preclude feasibility of attackers finding a solution by trial and error.

We finally consider the formal goals of the protocol. Günther [13] informally states that "the two parties construct keys which agree if they are both legitimate and do both conform to the protocol. The actual authentication is established when the decryption of the message sent by the other party is meaningful". Any demonstration

TABLE 3 Günther protocol (concrete)

A		B
T's signature (r_A,s_A) on h(A); s_A secret	$\rightarrow$ A, r_A	T's signature (r_B,s_B) on h(B); s_B secret
generate random x, compute $(r_B)^x$	B, r_B, $(r_A)^y$ $\leftarrow$	generate random y, compute $(r_A)^y$
compute $\overline{R}_B = \alpha^{h(B)} \cdot u^{-r_B}$	$\rightarrow$ $(r_B)^x$	compute $\overline{R}_A = \alpha^{h(A)} \cdot u^{-r_A}$
compute $K = (r_A^{\,y})^{s_A} \cdot \overline{R}_B^{\,x}$		compute $K = (r_B^{\,x})^{s_B} \cdot \overline{R}_A^{\,y}$

Since (A, r_A) and (B, r_B) are constant across protocol runs (as well as $\overline{R}_A$, and $\overline{R}_B$, for fixed parties A and B), if these are known a priori then the protocol may be reduced from three to two message exchanges. In this case, the protocol more closely resembles the Goss protocol, and can be made more similar if the multiply " ·" in the computation of K is replaced by an exclusive or ($\oplus$).

7.1 Formal analysis of Günther protocol

We first idealize the protocol. $R_A = (r_B)^x$ and $R_B = (r_A)^y$ are viewed as uncertified time variant keys of A and B, respectively. $\overline{R}_A = (r_A)^{s_A}$ and $\overline{R}_B = (r_B)^{s_B}$ are idealized as fixed public key-agreement keys[2] of A and B. These four quantities are then analogous to those of the same names in the Goss protocol of Section 6. A certificate

[1]The protocol can be made independent of the ElGamal signature scheme, by using any suitable alternate method to generate a pair (r, s), where r is a public key, s a secret key, and r^s publicly recoverable.

[2]These might alternatively be viewed as fixed "public identity keys" rather than fixed "public key-agreement keys".

[3]While related to the AT/GNY idea of "computable" possessions, this differs in that data is not only computed, but the result is idealized; the idealization of data in this protocol might be referred to as implicit signatures or implicit certificates.

of knowledge of the key (without compromising it) would serve equally well. “Actual authentication” is thus *not* part of the Günther key exchange per se. The intended formal goal is the same as that of Goss, namely secure key establishment (G3): A $\models$ A $\overset{K}{\leftrightarrow}$ B. The Günther protocol “has the advantage to generate a different key at each session” [13]; this is goal (G5). It was noted that “Proving the security of this scheme seems to be outside the scope of today’s methods”, and “the security could not be assessed within the current terminology” (p.32 and 36 resp. in [13]). These statements remain true, because the conclusions of logic analysis rely on the robustness of underlying algorithms. Nonetheless, given this, logic analysis establishes meaningful results about protocol security. We now outline these.

Lemma 9 The Günther protocol provides secure key establishment, i.e. achieves goal (G3).

Proof: [*Outline only*] Given the idealized form of the protocol, a proof analogous to that of Lemma 7 is as follows. After A receives R_B in (M7), as noted above A can compute B’s identity public key $\overline{R}_B$: A *has* $\overline{R}_B$. The semantics of the protocol lead A to conclude that this is B’s public key-agreement key. A also has enough information to compute a joint key, denoted K, by R30: A *has* K (see TABLE 3). To this point, our reasoning has established no properties of K; it is unqualified. Liberalizing the BAN symbol *sees* to include “computes from available information” (i.e. using it interchangeably with the AT/GNY *has*), we derive

$$A \textit{ sees } \{ PK_\delta(B, \overline{R}_B) \}_{sT} \tag{17}$$

Using (A1) for authenticity of T’s public key $K_T = u = \alpha^v$, and (A2) which allows A to trust the public key $\overline{R}_B$ computed from B, r_B and u, (17) yields, by R13,

$$A \models T \textit{ said } PK_\delta(B, \overline{R}_B) \tag{18}$$

“Verification” of the signature in (17) may include verifying the current validity of T’s public key u used in computing $\overline{R}_B$, and checking for revoked certificates.[1] These considerations are taken into account by (A10) which strengthens (18) to A $\models$ T $\models$ $PK_\delta(B, \overline{R}_B)$. This and (A4) with R3 yields A $\models PK_\delta(B, \overline{R}_B)$. Combining this with A’s belief in the quality of the private key-agreement keys of both A and B — (A5) and (A7) — R31 then provides: A $\models$ A $\overset{K}{\leftrightarrow}$ B. That is, upon protocol completion, A believes that K is shared with no party other than possibly party B. Here the fixed (certified) public key $\overline{R}_B = (r_B)^{s_B}$ plays the role of B’s public key-agreement key in R31, A’s fixed secret key s_A plays the role of A’s private key-agreement key, and the uncertified time-variant keys (x and R_B) are secondary parameters for the key agreement function *f*.

❑

Lemma 10 The Günther protocol provides key freshness, i.e. achieves goal (G5), provided B does not choose $y \equiv 0 \pmod{p-1}$.

Proof: Similar to proof of Lemma 5.

❑

8 Comparison of formal assumptions and goals

The formal analysis of the three protocols above allows a meaningful comparison to be made of their assumptions (TABLE 4) and guarantees (TABLE 5) below. These tables highlight the fact that the Günther and Goss protocols are identical with respect to formal goals, and very similar with respect to formal assumptions. The Günther protocol makes use of an identity-based scheme to authenticate the Diffie-Hellman public key r^s, whereas Goss uses explicit certificates to ensure their authenticity. The Günther protocol requires additional trust in the trusted party not to divulge user-specific secret keys. In Günther, with R_B and $\overline{R}_B$ as in Section 7, the key computed by A can be written $K = (\overline{R}_B)^x \cdot (R_B)^{s_A}$; i.e. B’s fixed certified key-agreement key powered by A’s uncertified time variant secret, times B’s uncertified time-varying exponential powered by A’s fixed certified secret key. Directly analogous in Goss with R_B and $\overline{R}_B$ as in Section 6, the key computed by A can be written $K = (\overline{R}_B)^x \oplus (R_B)^{\overline{x}}$; i.e. B’s fixed certified exponential powered by A’s time variant uncertified secret, combined with B’s time variant exponential powered by A’s fixed certified secret. It is also interesting to note that a neutral-party view of the Günther key is as $K = (\overline{R}_B)^x \cdot (\overline{R}_A)^y$.

TABLE 4 Comparison of formal assumptions

Assumption description[1]	STS	Goss	Günther
integrity of T’s public key	(A1)	(A1)	(A1)
quality of T’s private key	(A2)	(A2)	(A2)
control of binding sig. key	(A3)	—	—
quality of own k.a. key	(A5)	(A5)[2]	(A5)[3]
quality of other’s sig. key	(A6)	—	—
quality of other’s k.a. key	(A7)	(A7)[4]	(A7)[5]
control of binding k.a. keys	(A8)	(A4)	(A4)
freshness of own nonce	(A9)	(A9)	(A9)
ability to validate certificates	(A10)	(A10)	(A10)

[1] T=trusted authority; k.a.=key agreement key; sig.=signature
[2] required for both A’s fixed key $\overline{x}$ and variant secret x
[3] required for both A’s fixed secret s_A and variant secret x
[4] required for both B’s fixed key $\overline{y}$ and variant secret y
[5] required for both B’s fixed secret s_B and variant secret y

TABLE 5 Comparison of formal goals

Formal Goal	STS	Goss	Günther
far-end operative (G1)	yes	—	—
entity authentication (G2)	yes	—	—
secure key establishment (G3)	yes	yes	yes
key confirmation (G4)	yes	—	—
key freshness (G5)	yes	yes	yes
mutual belief - shared key (G6)	yes	—	—

The assumptions from Section 4 not required in the Goss and Günther protocols are (A3) and (A6) — since individual parties do not have their own signature key pairs — and (A8), replaced by (A4). However, as noted in the table, Goss and Günther require (A5) and (A7) twice each.

Consider now the goals of the Goss (and Günther) protocols relative to those of STS. (Comments about Goss apply equally to Günther). Goss results in the creation of a shared secret key which can be known to no one else aside from the intended party B, but does not provide proof of aliveness (G1); there is no freshness evident from the far-end’s message. Goss does not provide entity authentication in the sense of (G2); it is not evident that B’s message is either targeted to a specific party, or in response to a spe-

[1] However, true signature verification, or recognizability of a correct signature in (17), is not possible in this identity-based scheme. Instead, it is implicit: if the signature is invalid, the parties will not derive the same key K. This subsequent key confirmation is beyond the scope of the protocol as specified.

cific challenge. Finally, the Goss protocol does not set out to provide key confirmation (G4). While this allows an intruder to replay old messages and "complete" a fraudulent protocol run, fooling another principal into believing the run was successful, this is not a serious threat in practice as it provides no real advantage as an intruder cannot compute the resulting key, as will be evident once key usage commences. These missing goals can be easily provided in the Goss (or Günther) protocol by an additional message employing the established key, e.g. via encryption or a MAC.

The above analysis also allows us to compare these protocols with the X.509 two-way authentication protocol previously analyzed using BAN [11]. Due to space limitations here, the reader is referred to [11] for a description of the protocol, the formal assumptions it requires ($\alpha 1$ through $\alpha 7$), and the formal goals (Γ_1 through Γ_{12}). The X.509 three-way authentication protocol is more closely related to the above than the two-way protocol; the major difference is the use of timestamps (two-way) vs. random numbers (three-way). Both may be modified to accomplish Diffie-Hellman key agreement, although this was not their original purpose; in the three-way protocol, this may be done by replacing the X.509-specified "non-repeating numbers" r_A and r_B with Diffie-Hellman exponentials as in the protocols of the present paper.

Two assumptions in the X.509 logic analysis ($\alpha 5$ and $\alpha 6$) require that parties believe in the freshness of their *opponent's* timestamps and are able to check freshness in practice; the latter requires synchronized and secure time clocks. This is more demanding than (A9) above, which requires belief in the freshness of *self-generated* nonces. The X.509 analysis in [11] reflects an alternative to (A10) for handling public key distribution and checking the current validity of certificates in actual systems. "Duration stamps" and the ensuing requirement of $\alpha 7$ (assumption that the trusted party will not deliver certificates with invalid duration periods) were introduced to handle certificates having lifetimes spanning across protocol runs. The protocols analyzed in the current paper avoid use of timestamps, and thus certificate analysis necessarily differs. Aside from these differences, and the handling of certificates in X.509 as discussed above obviating (A4) and (A8), the X.509 analysis shows formal assumptions analogous to those in TABLE 4.

Regarding formal goals of the X.509 protocol, (G1) is attained, as is a goal similar to (G2) regarding entity authentication (namely Γ_5). A goal related to (G3) regarding secure key establishment was intended (Γ_{11}), but formal analysis revealed a technical problem in reaching this goal [11] (and thus (G6) also). Key confirmation (G4) was not intended as an original X.509 goal, nor was key freshness (G5), although the latter follows from Diffie-Hellman key agreement.

A comparison of the number of message exchanges required in the various protocols, excluding initial exchanges required for parties to acquire their own certificate, is given in TABLE 6. As implied by their names, the X.509 two- and three-way protocols require 2 and 3 messages, respectively; each requires one or more additional messages if the optional X.509 encryption field is utilized to exchange encrypted data.

TABLE 6 Comparison of number of messages

STS	Goss	Günther
3	2	3 [1]

[1] Can be reduced to 2 if fixed information is known a priori.

9 Concluding remarks

Several extensions and refinements applicable to BAN-like logics have been proposed to facilitate examination of beliefs and goals in authenticated key agreement protocols. Analysis using the extended logic has allowed direct comparison of the assumptions and goals of four authentication protocols. This highlighted the similarities between the Goss and Günther protocols, and qualitative differences between protocols providing key confirmation (e.g. STS) and those giving secure key establishment with implicit authentication (e.g. Goss, Günther).

While the most obvious objective of this method of formal analysis is to establish whether specified goals are achieved, this is only one of many benefits. The exercise forces one to identify, and express in precise detail, these goals; this is important in the absence of a universal definition of authentication. It also forces one to explicitly record the precise assumptions under which a protocol must operate. Furthermore, the exercise may in some cases result in more accurate specification of a protocol itself, as it requires detailed consideration of all protocols steps. For these reasons, and to allow meaningful comparisons, we feel there should be an onus on protocol designers to provide the results of such analysis concurrent with the proposal of a new protocol.

While many of these benefits might equally be achieved without logic techniques, the formality is a useful tool providing a template to follow, and a vocabulary for precise discussion of assumptions and goals. However as noted elsewhere, we emphasize these techniques do not (yet) provide an "automated theorem prover"; while the proofs in the logic themselves follow easily once a protocol is idealized and the requisite assumptions and goals are specified, the critical steps of capturing the assumptions and goals, and idealizing the protocol, do not appear amenable to automation simultaneously, and themselves require proof of correctness. Nonetheless, recent advances by others that allow automation, or partial automation, of one or more of these stages are encouraging.

Acknowledgments

Conversations with and/or comments from Li Gong, Lynn Marshall, Rainer Rueppel, Paul Syverson, Michael Wiener, and anonymous referees are gratefully acknowledged.

References

[1] M. Abadi, M. Tuttle. "A semantics for a logic of authentication". *Proc. 1991 ACM Symp. on Principles of Distributed Computing,* 201-216.

[2] R. Bird, I. Gopal, A. Herzberg, P. Janson, S. Kutten, R. Molva, M. Yung. "Systematic design of two-party authentication protocols". *Advances in Cryptology — CRYPTO'91, Lecture Notes in Computer Science* **576**, J. Feigenbaum (ed.), Springer-Verlag, 1991, 44-61.

[3] M.Burrows, M.Abadi, R.Needham. "A logic of authentication". *ACM Trans. Computer Systems* **8** (Feb. 1990), 18-36.

[4] M. Burrows, M. Abadi, R. Needham. "A logic of authentication". Digital Systems Research Centre, SRC Report #39 (1990 Feb. 22).

[5] CCITT Blue Book, Recommendation X.509: The Directory — Authentication Framework (1988). Also ISO 9598-4.

[6] W. Diffie, M. Hellman. "New directions in cryptography". *IEEE Transactions on Information Theory*, vol. **IT-22** (1976), 644-654.

[7] W. Diffie, P. Van Oorschot, M. Wiener. "Authentication and authenticated key exchanges". *Designs, Codes and Cryptography* **2** (Jun. 1992), 107-125.

[8] T. ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms". *IEEE Transactions on Information Theory*, vol. **IT-31** (1985), 469-472.

[9] V. Gligor, R. Kailar, S. Stubblebine, L. Gong. "Logics for cryptographic protocols — virtues and limitations". *Proc. IEEE 1991 Computer Security Foundations Workshop* (Franconia, New Hampshire).

[10] L. Gong, R. Needham, R. Yahalom. "Reasoning about belief in cryptographic protocols". *Proc. 1990 IEEE Symp. on Security and Privacy* (Oakland, California), 234-248.

[11] K. Gaarder, E. Snekkenes. "Applying a formal analysis technique to the CCITT X.509 strong two-way authentication protocol". *J. Cryptology* **3** (Jan. 1991), 81-98.

[12] K.C. Goss. Cryptographic method and apparatus for public key exchange with authentication. U.S. Patent 4,956,863 (granted 1990 Sept. 11).

[13] C. Günther. "An identity-based key-exchange protocol". *Advances in Cryptology — EUROCRYPT'89, Lecture Notes in Computer Science* **434**, J.-J. Quisquater and J. Vandewalle (eds.), Springer-Verlag 1990, 29-37.

[14] R. Kailar, V. Gligor. "On belief evolution in authentication protocols". *Proc. IEEE 1991 Computer Security Foundations Workshop* (Franconia, New Hampshire), 103-116.

[15] W. Mao, C. Boyd. "Towards formal analysis of security protocols". *Proc. Computer Security Foundations Workshop VI* (Franconia, New Hampshire, June 1993), 147-158, IEEE Computer Society.

[16] D.M.Nessett. "A critique of the Burrows, Abadi and Needham logic". *Operating Systems Review* 24 (1990), 35-38.

[17] C.P.Waldvogel and J.L.Massey. "The probability distribution of the Diffie-Hellman key". Presented at AUSCRYPT'92 (Dec. 1992).

Appendix A: Authentication logic background

This section reviews the BAN logic [3] including refinements by GS [11]. Proofs are constructed in the logic by a four-stage process. First the protocol is "idealized" — the actual or concrete protocol is expressed as a sequence of formal steps (A → B: X) where A and B are the communicating entities and X is a statement in the syntax of the logic. Second, the assumptions under which the protocol operates are identified and formally expressed. Third, the goals of the protocol are identified and formally expressed. Finally, a proof is constructed showing that given the basic assumptions, upon observing the proper protocol messages the parties involved are able, through the logical postulates (see below), to arrive at a state where they believe the formal goals. The nature of the logic analysis depends heavily on the precise details of the formalization of initial assumptions, the idealization of the protocol, and the formalization of goals. Unfortunately, transformation of the protocol into an idealized form cannot itself be automated, nor proven to be correct. For these reasons, the idealization is recognized as the most critical step.

We first review the basic notation — the logic symbols and their informal semantics. A and B are parties (principals) involved in the protocol, X is a statement, and K is a cryptographic key.

A *once_said* X — A once sent the message X. This could have been in either the current protocol run or a previous run. In BAN it is understood that a principal only says things which he believes.

A $|\equiv$ X — A *believes* X (or is entitled to believe X). If X is a data value rather than a statement, "A believes X" is best interpreted as "A believes X is true" or "A said X in the current epoch".

A *controls* X — A *has jurisdiction over* X; A can be trusted on this matter. If you believe that A believes X, and if you trust A on X, then you can believe X also.

A *sees* X — A observes message X. A *sees* X if X arrives as a protocol message.

A $\overset{K}{\leftrightarrow}$ B — A *shares the good key* K with B. The key is suitable for use as a cryptographic key in that only A and B know it, it will not be disclosed to others, and it can not be deduced by others.

fresh(X) — X is recent, and has not been seen prior to its present use. Time is broken into two periods: the present (the current epoch, beginning with the start of the current protocol run) and the past. X is fresh if it is not the replay of a message from the past. Formulas generated specifically for the purpose of being fresh are called *nonces*.

In the protocols examined in the present paper, so-called "random numbers" serve as nonces. The critical properties are that they be unpredictable, and drawn from a sufficiently large space, with vanishing probability of repetition. Provided such numbers are not intentionally re-used, and are generated using an appropriate number generator, the probability that such a number duplicates a previous such number is vanishingly close to zero, and for practical purposes can be assumed equal to zero.

PK(K,A) — A *has associated with it the good public key* K. The key is good in the sense that K is A's authentic public key, and there exists a unique (public, private) key pair corresponding to K.

$\prod$(A) — A *has associated with it some good private key.* The key is good in the sense that it is known by no one else, nor can it be deduced by anyone else.

R(A, X) — A *is the intended recipient* of X.

The last three constructs are from GS. BAN uses a construct similar to PK(K,A) with semantics "A has K as a public key", implicitly defining a corresponding private key K^{-1} never discovered by anyone aside from A. To embrace the idea of a signed message and an encrypted message, we use the following notation and semantics (notation for the first differs from BAN):

$\{X\}_{sA}$ — The *signature* of A on X, using A's private signature key. Note that X is not in general recoverable from $\{X\}_{sA}$, depending on the type of signature mechanism used and the possible hashing before signing.

$\{X\}_K$ — Data resulting from *encipherment* of X under symmetric key K, with a fixed symmetric encipherment algorithm assumed. Where relevant (e.g. R1 below), this is short for $\{X\}_{K \text{ from } R}$, and in BAN it is assumed a party can distinguish its own enciphered formulae from those generated by other parties R.

BAN logic establishes beliefs a party is entitled to when all protocol steps are successful. In proofs, B *sees* $\{X\}_{sA}$ should be taken to mean B has received a message containing a data item Y, and has verified the format of Y to be that of a signature on X using a key corresponding to the public signature key which B associates with A. It is implicitly assumed that B possesses this public key, and that there is sufficient information available, or redundancy in X, to allow signature verification. Such verification itself appears in BAN only implicitly (e.g. see R13 below). Similarly, A *sees* $\{X\}_K$ is taken to mean A has received a message containing a data item Y, and has verified the format of Y to be that of the encryption of X under key K; again it is assumed there is sufficient a priori knowledge or redundancy to allow verification that K is the correct key, and verification itself appears only implicitly (e.g. R1 below). The GNY "recognizability" construct (see Section 2) addresses this explicitly.

For reference, and to put our work in perspective, we now list a subset of BAN inference rules previously proposed. The rules are logical postulates which allow proofs to be constructed. Of those below, all but R13 (which is from [11]) are from the original BAN logic; R1 through R13 are numbered as in [11] for cross-reference. The first rule R1 is read as follows: If A believes that A shares a good key K with B, and if A sees a message X encrypted under key K (which she herself did not encrypt), then A believes that B once said X.

R1. (Message meaning rule for shared keys)

$$\frac{A \mid\equiv A \overset{K}{\leftrightarrow} B, \quad A \textit{ sees } \{X\}_{K \text{ from } U}}{A \mid\equiv (B \textit{ once\_said } X)} \quad \text{where } U \neq A$$

R2. (Nonce-verification rule)[1]

$$\frac{A \mid\equiv \textit{fresh}\,(X), \quad A \mid\equiv (B \textit{ once\_said } X)}{A \mid\equiv (B \mid\equiv X)}$$

R3. (Jurisdiction rule)

$$\frac{A \mid\equiv (B \textit{ controls } X), \quad A \mid\equiv (B \mid\equiv X)}{A \mid\equiv X}$$

R4. (Belief aggregation)

$$\frac{A \mid\equiv X, \quad A \mid\equiv Y}{A \mid\equiv (X, Y)}$$

R5. (Belief projection)

$$\frac{A \mid\equiv (X, Y)}{A \mid\equiv X}$$

R6. (Mutual belief projection)

$$\frac{A \mid\equiv (B \mid\equiv (X, Y))}{A \mid\equiv (B \mid\equiv X)}$$

R7. (Once-said projection)

$$\frac{A \mid\equiv (B \textit{ once\_said } (X, Y))}{A \mid\equiv B \textit{ once\_said } X, \quad A \mid\equiv B \textit{ once\_said } Y}$$

[1]X must be fresh in R2 as in BAN a party is bound to its beliefs (only) for the duration of a single protocol run.

R10. (Sight projection)

$$\frac{A \textit{ sees } (X, Y)}{A \textit{ sees } X, \quad A \textit{ sees } Y}$$

R12. (Freshness propagation rule)[2]

$$\frac{A \mid\equiv \textit{fresh}\,(X)}{A \mid\equiv \textit{fresh}\,(X, Y)}$$

R13. (Message meaning rule for public signature keys)[3]

$$\frac{A \mid\equiv PK(B, K), \quad A \mid\equiv \prod(B), \quad A \textit{ sees } \{X\}_{sB}}{A \mid\equiv (B \textit{ once\_said } X)}$$

R21. (Message decryption rule for symmetric keys)

$$\frac{A \mid\equiv A \overset{K}{\leftrightarrow} B, \quad A \textit{ sees } \{X\}_K}{A \textit{ sees } X}$$

R22. (Message decryption rule for unqualified keys)[4]

$$\frac{A \textit{ has } K, \quad A \textit{ sees } \{X\}_K}{A \textit{ sees } X}$$

R23. (Hash function rule)

$$\frac{A \mid\equiv (B \textit{ once\_said } H(X)), \quad A \textit{ sees } X}{A \mid\equiv (B \textit{ once\_said } X)}$$

where H() is an appropriate hash function.

[2]R12 implies that if part of a formula is fresh, the entire formula is. Note a non-fresh formula cannot be made fresh by concatenating it to a fresh formula; here (X, Y) is a message unit whose integrity is protected, e.g. cryptographically.

[3]R13 assumes a message X can be recovered from a signature on it (i.e. signature with message recovery, no hashing) and requires possession of the corresponding public key. If the former is not possible, a pre-condition is A *has* X.

[4]Modified slightly from BAN, to make use of the GNY/AT "*has*" construct (see Section 2).