

Accountability in Electronic Commerce Protocols

Rajashekar Kailar
kailar@mindspring.com

Abstract— In most commercial and legal transactions, the ability to hold individuals or organizations accountable for transactions is important. Hence, electronic protocols that implement commercial transactions must be designed to provide adequate accountability assurances for transacting parties. Without such assurances, electronic transactions can be susceptible to disputes. Currently, protocol design for electronic commerce is done in an ad-hoc manner, a technique which has been shown to be error-prone by past experience with key distribution protocols [4]. Despite the importance of accountability in electronic commerce, and the subtlety of designing error-free protocols, currently, there are no analysis methods to examine whether a protocol design conforms to the accountability goals of the transaction that it implements. Since most current protocol analysis methods have been developed to analyze key management protocols, they focus on properties such as message replay detection, and key origin authentication (e.g., [20], [4], [11], [14], [18]).

In this paper¹, a new framework is proposed for the analysis of communication protocols that require accountability, such as those for electronic commerce. This framework can be used to analyze protocol designs to detect accountability (or lack thereof). Arguments are presented to show that a heretofore un-explored property “provability” is pertinent to examining the potential use of communication protocols in the context of litigation, and in the context of audit. A set of postulates which are applicable to the analysis of proofs in general and the proofs of accountability in particular, are proposed. The proposed approach is more natural for the analysis of accountability than the existing belief logics (e.g., [4]) that have been used in the past for the analysis of key distribution protocols. Some recently proposed protocols for electronic commerce and public-key delegation are analyzed to illustrate the use of the new analysis framework in detecting (and suggesting remedies for eliminating) their lack of accountability, and in detecting and eliminating redundancies.

I. INTRODUCTION

Electronic Commerce using wide area networks like the Internet has the potential to revolutionize the way many businesses are conducted in future. Using the Internet as a medium for conducting commercial transactions enhances the accessibility of a large customer base to a wide variety of information and services, and greatly enhances the ease of making remote payments. Consequently, there is growing interest in the design and development of electronic protocols for conducting commercial transactions in wide area networks [5], [6], [7], [16], [19], [21], [22], [23], [24]. Companies and consortia to promote business on the Internet are proliferating [30].

However, if electronic transactions on the Internet fail to provide the same assurance of accountability as the paper-world transactions that they intend to substitute, they can be susceptible to disputes between transacting parties.

Without adequate accountability assurances in electronic commerce transactions, there would be no means to reliably enforce punitive measures against fraudulent individuals or organizations². The proper design of electronic transactions can help eliminate lack of accountability.

Experience in the past with key distribution protocols has shown that protocol design is an error-prone task, and that formal analyses can aid in detecting and correcting weaknesses and redundancies in design stages. Most protocol analysis methods to date reason about the evolution of beliefs for the protocol participants as a result of the messages that are sent or received (e.g., [4], [11]), and are popularly referred to as *belief logics*.

To date, there is ample evidence for the success of belief logics in the analysis of key management protocols. However, as shown in this paper, the analysis of accountability can be done more naturally using a new framework that reasons about the ability of participants in a protocol *to prove* accountability. Although this paper demonstrates the use of this framework primarily by analyzing electronic commerce protocols, the proposed framework is just as applicable to other protocols that require accountability.

The proposed framework applies the principles that are derived from real life commercial and legal transactions to the analysis of accountability in protocols. In most commercial and legal transactions, the ability to hold individuals or organizations accountable for transactions is important. For instance, a signed receipt can be used to hold a vendor accountable for a product (e.g., if the product is defective), and the buyer accountable for the payment (e.g., if his check bounces). Accountability in some special transactions, such as business agreements, may require endorsements by trusted authorities, such as notaries. Similar principles hold in electronic commerce transactions, where the ability to unambiguously *prove* the association between a principal (user) and a statement (message) is essential if the corresponding paper-world transactions require accountability. We use these principles to formulate the goals of, and to analyze the properties of protocols for electronic commerce. A set of postulates is proposed for the analysis of accountability in protocols.

Herein, the use of the proposed framework in the analysis of accountability is illustrated by example. In the first example, one of the proposed protocol alternatives for Carnegie Mellon’s Internet Billing Server protocol [23], that uses only asymmetric encryptions, is analyzed using the postulates to show that it can achieve its desirable accountability properties using some assumptions and additional

¹A preliminary version of this paper appeared in the *Proceedings of the IEEE Symposium on Security and Privacy*, in Oakland, CA, May 1995 [13].

²The lack of accountability in ATM transactions is known to have caused numerous miscarriages of justice in settling disputes between fraudulent parties and their victims [25].

steps which are not required if some design modifications are done to the original protocol. The analysis also points to a redundancy in this protocol, the elimination of which makes this protocol more efficient. In the second example, an “optimized” version of the first protocol, which is described in [23], is shown to lack many desirable accountability properties, making the proposed protocol alternative useful only with strong assumptions. Using the postulates, design modifications are suggested to restore accountability in this protocol.

In the third example, USC ISI’s anonymous payment protocol [19] is analyzed to show that the protocol does not provide both the payor and the payee with some accountability assurances discussed in this paper, in agreement with the goals of this protocol. The fourth example illustrates the use of the proposed postulates in the analysis of public-key delegation protocols, by examining the SPX protocol [28].

The balance of this paper is organized as follows. In Section 2, the necessary background on related research work is provided. Section 3 presents informal discussions on *belief*, *provability* and *accountability*. Sections 4, 5 and 6 describe the analysis framework. Section 7 presents some examples of transaction goals, and Section 8 presents four examples to illustrate the application of the analysis framework. A brief discussion of the endorsement chains that may be required to establish accountability is presented in Section 9. Section 10 concludes the paper.

II. BACKGROUND

As a result of subtle design errors observed in numerous currently implemented communication protocols, and the relative difficulty of designing secure protocols, the need for formal analysis of protocols has been recognized in the past [29], [10], [27]. In response to this need, several analysis methods have been proposed to date [20], [4], [18], [11], [14]. Most formal methods proposed for protocol analysis use the *state machine* model in one form or another. That is, the attributes of the protocol participants are used to define the states, and the protocol message exchanges are viewed as state transitions.

For instance, the approach of Millen *et al* examines protocol specifications for insecure state transitions using an automated tool [20]. Analysis logics, such as the one proposed by Burrows *et al* model state transitions in a protocol as dialogues between real-life entities. In this context, protocol goals can be articulated in terms of the *beliefs* that should ensue for protocol participants at the end of the protocol run. Protocol analysis logics typically consist of postulates that define the evolution of these beliefs. A mismatch between the protocol goals and the derived results can be interpreted as a protocol weakness or a redundancy.

Logics for protocol analysis examine the properties of protocols based on the premise that the analyzed system adheres to a certain behavior. In other words, most logics function at a specific level of abstraction, making several assumptions about lower level mechanisms. The results derived using logics are, hence, dependent on the validity of

the assumptions in the environment of operation. The dependencies between the results derived and the properties that are assumed determines the effectiveness of protocols, and has been studied by Kailar *et al* [15].

Protocol analysis has focused on message (content) properties, such as key freshness, message confidentiality, and message origin authentication. With the advent of electronic commerce, cryptographic protocols are being adapted for implementing commercial transactions, and may need to provide accountability for protocol participants. Consequently, in electronic commerce protocols, the ability to prove the association of protocol messages with protocol participants needs to be analyzed. Existing analysis methods have not been designed to address this important property. The analysis approach proposed in this paper attempts to address this need.

Early work on electronic commerce and digital cash was pioneered by Chaum and others [5], [6], [7]. While the focus of their work was to provide low-level computational algorithms that meet the needs of electronic transactions, recent work on electronic commerce has focused on formulating client-server model based electronic commerce protocols which can be deployed on wide area networks, such as the Internet [19], [16], [23]. While these two directions of research are complimentary, the absence of formal analysis methods in both areas is conspicuous; this paper attempts to fill the gap in the analysis of the latter type of protocols.

III. BELIEF VS. PROVABILITY

This section makes a distinction between beliefs and proofs, in the context of accountability analysis. In simple terms, an individual is said to believe a statement if he is convinced of that statement³. An individual can prove a statement to another individual if he can convince the latter about the statement.

A proof of a statement x is a set of statements that can collectively convey the validity of x to an audience. The ability to prove a statement x can be considered to be the ability to produce the necessary set of statements that can convince the audience of x .

In practice, it may not be feasible to convey a self contained (i.e., a sufficient) set of statements which establishes a proof. This is because a self contained set of statements becomes prohibitively large for the proof of even the simplest of statements. For specialized, complex statements, self contained proofs become inconceivably large⁴. However, within the context of a specific system, whose semantics can be assumed to be common knowledge to the audience of a proof, it is possible to have a self contained set of statements which convey the validity of a statement to that audience.

³“According to classicists, when one believes a mathematical statement, one believes that in *principle* there is a correct, formal, valid, step by step, syntactically checkable deduction leading to x in a suitable logical calculus” [8].

⁴e.g., it is estimated that a formal demonstration of one of Ramanujan’s conjectures assuming set theory and elementary analysis would take about two thousand pages; the length of a deduction from first principles is inconceivable [8], [17].

Further, the sequence of steps that need to be executed to prove a statement x need not be unique⁵. The steps that are required to convince an audience may depend, in addition, on the initial state of the audience. In the initial state (before the execution of a proof), the audience is already assumed to know a set of facts which can influence the proof.

In the context of cryptographic protocol analysis, the notion of belief has been coupled with time. For instance, the logic of Burrows *et al* [4] makes the implicit assumption that a principal believes a statement if it uttered the statement recently. In contrast, proofs are, to a large extent, independent of time. For instance, in the absence of overriding proofs of forgery, it can be proved indefinitely, that an individual is accountable for a message that he signs on paper. Although proofs of accountability based on digital signatures do not have the same type of time independence that written signatures have (i.e., the digital signatures are computed with keys which are updated with time), this paper does not analyze the dependency on freshness. Instead, this paper analyzes the lack of accountability that may be built into protocol designs, despite the validity of digital signatures and the freshness of messages.

A. Accountability

Accountability is the property whereby the association of a unique originator with an object or action can be proved to a third party (i.e., a party who is different from the originator and the prover).

B. Applicability of Belief to Analyze Accountability

In the past, belief logics have been used to analyze the evolution of beliefs of participants in protocols. Using belief logics helps establish properties, such as the non-replay of session keys, which the principals participating in a secure communication session may be required to believe, but may not be required to prove to third parties.

Unlike key distribution protocols, however, in protocols for electronic commerce, principals may be required to prove to third parties (e.g., auditors or courts of law⁶) that transacting parties are accountable for the messages that they send. Reasoning using *beliefs* does not lend itself to the analysis of what each participant in a protocol *can prove* to third parties on completion of the protocol. Since the ability to reason about provability is important for the analysis of accountability, a framework for analyzing provability is more suitable for this purpose than the existing logics of belief.

While it may be possible to extend current logics of belief to include the analysis of provability, it is useful to maintain separation between the analyses of these properties, for simplicity, and since all protocols do not require both properties.

⁵For example, to date, over fifty proofs of Gauss's "law of quadratic reciprocity" are known [8].

⁶It is noteworthy that the acceptability of Digital signatures as proof of accountability in the evidentiary process remains to be tested [3].

C. Symmetric vs. Asymmetric Encryption

If Alice and Bob share a secret key, and if Bob receives a message encrypted with this key, he can *believe* that Alice sent this message⁷ (if he has not sent it himself), but cannot *prove* this to a third party (e.g., a court of law), unless Bob is trusted by the third party not to fake a message using the shared key and hold Alice responsible for that message. However, such an assumption of trust would not be necessary if Alice were to digitally sign the message, either using her private key of an asymmetric key pair, or using other strong digital signature algorithms which allow signature authentication. Symmetric key encryption schemes do not allow us to hold principals accountable for statements unambiguously without requiring additional assumptions of trust.

If all transactions are mediated by a trusted server, the lack of accountability in shared key systems may not be perceived as a major problem in certain systems. This is because, in this case, messages that require accountability may be encrypted with keys that clients share with a trusted server. Since the server is trusted, it can be assumed not to masquerade as clients.

In the context of litigation, however, trusted servers do not, and probably cannot exist. Hence, the statements made about accountability in protocols that depend on trusted servers (e.g., [23]) are questionable, at best. Further, in systems using trusted servers, the requirement that all transactions should be mediated by a trusted server places a major responsibility on the trusted server [26]. A security compromise of the trusted server can have dire consequences.

Although asymmetric encryption schemes provide accountability using fewer assumptions than symmetric schemes, they too, are not entirely tamper-proof. A principal who signs a statement could attempt to repudiate his own statement by intentionally compromising his private key⁸. This fraud can be countered in part by notarization, which shows the time a statement was made (i.e., that it was made before the key was compromised). However, this solution would require that, in the absence of overriding proof for the time of key compromise, principals be held accountable for any messages signed with their keys till the key compromise is reported to authorities.

IV. ANALYSIS FRAMEWORK

In this paper, the ability of principals to prove message (statement) origin in communication protocols is analyzed. The analyzed system consists of a group of users or user clients, which we call *principals*, which send messages to each other. Principals are denoted by upper-case letters $\{A, B, \dots\}$. A signed message functions as a statement made (non-repudially) by the principal who signs the message. The statement made by each message is the message interpretation, and is expected to be defined by protocol designers. Statements are denoted by lower-case letters $\{x, y, \dots\}$.

⁷This is the gist of the *Message Meaning Rule* of [4].

⁸I am grateful to Clifford Neuman for pointing this out.

Within this framework, a proof of a statement x is something which convinces another principal of statement x . This paper does not attempt to define the specific steps a principal needs to convince itself of a statement. This is because, these steps are largely dependent on the type of statement, the originator of the statement, the principal(s) to whom it is being proved, and on the system on which the statement is made⁹. Rather, we define the conditions under which a proof of a statement can be conveyed effectively to another principal.

Depending on the ability of the prover to prove x , there can be at least two types of proofs:

- *Strong Proof*: “ A CanProve x ”

Principal A can prove statement x if, for any principal B , A can execute a sequence of operations such that after the sequence of operations, B is convinced of x without revealing any secret y ($y \neq x$) to B .¹⁰

Although the above definition requires A to be able to prove x to all possible principals, in this paper, the above definition is interpreted more loosely, as A being able to prove x to all principals in the intended audience of a proof. There are (at least) two degrees to which A can prove x to B .

- *Transferable proof*: A is said to provide a *Transferable proof* of x to B if A CanProve x to principal B such that after the proof, B CanProve x .
- *Non-transferable proof*: A is said to provide a *non-transferable proof* of x to B if A CanProve x to B such that after the proof, B CanProve x to B (see definition below of the CanProve to construct) i.e., B is convinced of x after A has proved x to it, but may not necessarily be able to convince some other principal, say C , that x holds. Examples of non-transferable proofs of accountability (identity) are Zero-Knowledge signature schemes (e.g., [12]).
- *Weak Proof*: “ A CanProve x to B ”

In this type of proof, the prover has the capacity to prove x to a specific principal B without revealing any secret y ($y \neq x$) to B . Weak proofs can be *transferable proofs* or *non-transferable proofs*. The definitions of these two types of proofs are similar to those provided for *Strong proofs*, and will not be repeated.

Trivially, a principal who can provide a *strong proof* can provide a *weak proof*. The proofs described in this paper pertain to a specific system, namely a network which consists of principals who make statements to conduct electronic transactions. Hence, the *Strong Proofs* of this paper are, still, relative to accountability in communication protocols, and hence, assume

⁹e.g., convincing oneself to buy an expensive painting based on its genuineness may require different steps depending on the buyer, the seller, and the value of the painting.

¹⁰The secrecy of y is necessary for the use of this construct within the context of Zero Knowledge Proofs, and in the **Sign** postulate, which is explained in Section 6(B). Note that the requirement of secrecy may be too restrictive for some cases, such as proof in shared key systems, wherein the shared key may need to be revealed to the audience.

knowledge of its associated properties. An example of a protocol that is based on weak proofs is described in [21].

- *Signature Authentication*: “ K Authenticates A ”

This construct is used to denote the fact that key K can be used to authenticate the signature of principal A , or to associate principal A unambiguously with any statement encrypted with K^{-1} . Here, K and K^{-1} are public and private counterparts of an asymmetric key pair.

Statements on signature authentication (i.e., of the form K Authenticates A) are used in this analysis to interpret public key certificates, which are issued by certifying authorities.

- *Message Interpretation*: “ x in m ”

x is an interpretation of a field, or a combination of fields in message m . This interpretation is protocol specific, and is expected to be defined by the protocol designers explicitly. Decrypted values of encrypted message fields cannot be used in the interpretation of x unless the decrypted values are also fields in the signed message.

Attributes, such as freshness, are made explicitly as part of the statement itself. In other words, x may be of the form (y is valid until T). Hence, x is defined entirely by the interpretation of the fields within the signed message m .

- *Statement*: “ A Says x ”

That is, principal A is accountable for making the statement x , and anything that x implies. This is used for interpreting provability results of the form A CanProve B Says x . The postulate

$$A \text{ Says } (x, y) \Rightarrow A \text{ Says } x$$

is used implicitly in the analysis. In other words, if principal A says any statement which is composed of two facts x and y , then A is also responsible for saying each facts x individually.

- *Message Receipt*: “ A Receives m SignedWith K^{-1} ”

The principal A receives message m which is signed with K^{-1} . m here includes all contents and signatures associated with the message. The postulate

$$\frac{A \text{ Receives } m \text{ SignedWith } K^{-1}; x \text{ in } m}{A \text{ Receives } x \text{ SignedWith } K^{-1}}$$

is used implicitly in the analysis. Here, x is an interpretation of a field, or of a combination of fields within a message.

- *Trust*: “ A IsTrustedOn x ”

This denotes the fact that principal A is trusted on statement x . In particular, A has the authority to endorse statement x , and is liable for making statement x .

In this analysis, statements of the form A IsTrustedOn x are used to articulate the trust assumptions on key certifying authorities, or to interpret authorization certificates that are issued by authorities. Two degrees of

trust are defined¹¹ :

– *Global Trust* :

If A is globally trusted, then A `IsTrustedOn` x by all principals B ¹². In this analysis, although a principal A may not necessarily be globally trusted on x in the strict sense, the fact that A is trusted by the intended audience for a proof of x is considered sufficient to treat A as a globally trusted authority of x .

– *Non-global Trust* :

In this case, A `IsTrustedOn` x by B . This definition applies to a specific principal B to whom the proof is being delivered. All postulates that are articulated in this paper using *Strong proof* and *Global trust* definitions can be applied to examples with *Weak proofs* and *non-global trust* definitions. This definition can also be useful if the postulates need to be applied to a specific principal B (e.g., A specific District Attorney’s office), to whom a statement needs to be proved.

Each of these definitions can be qualified, in addition, by adding context, such as domain, configuration or time. For instance, statements like A `IsTrustedOn` x in *domain_A*, K `Authenticates` A until T are conceivable.

V. ANALYSIS ASSUMPTIONS

The protocol weaknesses that are detected using the proposed analysis framework are inherent to the design of the protocols, and hence, do not require any assumptions to hold. That is, these weaknesses would remain regardless of whether or not the stated assumptions hold. However, the accountability assurances that are derived using this analysis framework are based on the premise that the following assumptions hold. The lack of support for the assumed properties in the environments of operation can render the assurance results of the analysis invalid. The assumed properties are :

- *Digital Signature Algorithms* :

The digital signature algorithms discussed in this paper are based on the public-key encryption paradigm introduced by Diffie and Hellman [9]. Signature algorithms are assumed to be robust enough so that the signatures can be undisputably associated with individual users. It is assumed that the signature algorithms are computationally unbreakable for a sufficiently long time. Digital signatures are assumed to provide message origin authentication, message content integrity and message sender non-repudiation. Unlike the signature scheme described by Chaum and van Antwerpen in [6], verification of digital signatures is assumed not to require the consent of the signer.

- *Trust* :

Principals are trusted not to willingly share their private keys with anyone for whom they are not willing to be accountable. Since principals are liable for

preserving the privacy of their own private keys, it is in their own interest to preserve their key privacy. In symmetric key schemes, to establish accountability, principals need to be trusted not to masquerade as other principals with whom they share keys. This is a stronger assumption than that which is required in asymmetric systems.

This paper also assumes that in the absence of an entity (or group of entities) who is (are) trusted by the audience on the statement being proved, the prover cannot successfully convince the audience about the validity of a statement¹³. In other words, a principal to whom a statement is being proved can be convinced of the statement only if either (1) it is an authority on the verification of the statement, or (2) it can obtain an endorsement of a trusted party (or parties) on the validity of the statement. In the absence of a globally trusted party, this requires the existence of a global trust hierarchy.

- *Message Integrity* :

Like all other analysis methods which operate at a relatively high level of abstraction, we assume that the message integrity is preserved. In particular, it is not possible (or, possible with negligible probability) to fake a signed message using parts of other messages or to compute a private key and to sign a message such that the signature is mistakenly accepted as authentic.

- *Availability of Services*:

The statement A `CanProve` x assumes that at some time in the future, A has the ability to send the required set of messages to prove x . We assume that if there are denial of service problems which keep A from sending the messages to prove x , they will eventually be eliminated such that the service (e.g., network connectivity) is made available to A so that it can prove x .

- *Certificate Revocation*:

In this analysis, it is assumed that the revocation of public key certificates and authorization certificates is enforced such that the original certificates are not usable in proving new statements that are made immediately thereafter. That is, proofs of statements that use revoked public key (authorization) certificates are valid only if the statements were signed (endorsed) when the corresponding certificates were valid. This assumption removes considerations of freshness from the subject matter of this analysis. In practice, this property is enforced by maintaining up to date publicly viewable certificate and key revocation lists.

VI. POSTULATES

In this section, we use the notation introduced to articulate some properties of accountability. The properties are divided into generic properties (i.e., those that can be applied to proofs in general), and properties which are specific to accountability in electronic communication protocols.

¹¹I am grateful to Paul Syverson for pointing out the two degrees of trust.

¹²Examples of global trust is the trust placed on the Treasury Department’s currency denominations.

¹³Note that this trusted entity can belong to the audience of the proof.

A. Generic Properties of Provability

The **CanProve** construct has several generic properties which are useful for the propagation of proofs. Below, some of the generic properties are postulated. The symbolic presentation of these postulates is similar to that of [4]; i.e.,

$$\frac{P ; Q}{R}$$

means that if statements P and Q hold simultaneously, then statement R holds. The use of ‘ $;$ ’ denotes *conjunction* and not temporal ordering.

- **Conjunction**

This postulate defines conjunction of proofs :

$$\text{Conj: } \frac{A \text{ CanProve } x ; A \text{ CanProve } y}{A \text{ CanProve } (x \wedge y)}$$

That is, if A can prove that x holds and A can prove that y holds, then A can prove that $x \wedge y$ holds. This postulate can be used (1) to examine what each principal can prove by composing the individual statements that it can prove, (2) to hold principals accountable for composite statements from the individual statements for which they can be held accountable and (3) to show inconsistencies in a principal’s statements.

- **Inference**

This postulate relates provability to inference.

$$\text{Inf: } \frac{A \text{ CanProve } x ; x \Rightarrow y}{A \text{ CanProve } y}$$

That is, if a principal A can prove that x holds, and if x implies y , then it follows that A can prove that y holds. This postulate is used in the analysis to derive implied results from statements that are proved.

In the analysis, the statement $(x \Rightarrow y)$ is used to articulate the interpretations of signed messages. Such interpretations are assumed to be defined explicitly by protocol designers, and hence, are assumed to be evident to all principals involved.

- **Relationship to Belief**

The following postulate describes the relationship between the notions of *provability* and *belief*¹⁴.

$$(A \text{ Believes } x) \Leftrightarrow (A \text{ CanProve } x \text{ to } A)$$

That is, A believes x if and only if A can prove to itself that x holds. In particular, A may prove x to itself by passing x through a set of checks, or based on an endorsement of an authority of x .

This relationship is not used in this paper, since the analysis that is presented does not require the use of beliefs. This is provided here primarily to establish relationship with a notion (belief) that has been researched for the past several years.

- **Relationship of Strong and Weak Provabilities**

The postulates in this paper are articulated using

strong proofs. However, these postulates are also applicable to the analysis of weak proofs. The following postulate relates strong and weak proofs, and can be used to extend the axioms for the analysis of weak proofs.

$$\frac{(S ; C \text{ CanProve } y) \Rightarrow (A \text{ CanProve } x)}{(S ; C \text{ CanProve } y \text{ to } B) \Rightarrow (A \text{ CanProve } x \text{ to } B)}$$

That is, given that a set of statements S holds, if A can prove x provided C can prove y , then, under identical conditions (i.e., S), A can prove x to B provided C can prove y to B . Here, $C(y)$ is not necessarily different from $A(x)$.

- **Relationship of Global and Non-Global Trust**

The postulates in this paper are articulated using global trust assumptions. However, these postulates are also usable with non-global trust assumptions. The following postulate relates global and non-global trust assumptions, and can be used to extend the axioms for the analysis of weak proofs.

$$\frac{(S ; C \text{ IsTrustedOn } y) \Rightarrow (A \text{ CanProve } x)}{(S ; C \text{ IsTrustedOn } y \text{ by } B) \Rightarrow (A \text{ CanProve } x \text{ to } B)}$$

That is, given that a set of statements S holds, if A can prove x provided C is trusted on y , then under identical conditions (i.e., S), A can prove x to B provided C is trusted on y by B . Here, $C(y)$ is not necessarily different from $A(x)$.

B. System Specific Provability Properties

This section describes the properties of the **CanProve** construct that are specific to the system being analyzed. In particular, the postulates below summarize the properties of systems that use digital signatures, and relate the provability properties to the trust on principals (e.g., certifying authorities) to make specific statements.

- **Accountability Property of Digital Signatures**

When digitally signed, messages serve as binding statements for which the principals who sign the messages are accountable. This accountability can be proven by any principal who can prove the association between the signature on the message and the signing principal. The postulate is as follows :

$$\text{Sign: } \frac{A \text{ Receives } (m \text{ SignedWith } K^{-1}) ; x \text{ in } m ; A \text{ CanProve } (K \text{ Authenticates } B)}{A \text{ CanProve } (B \text{ Says } x)}$$

That is, if A receives a message m which is signed with key K^{-1} , the message m contains statement x , and if A can prove that the key K authenticates principal B (or that K authenticated B at the time the message was signed), then A can prove that B indeed says x . Here, the implicit assumption is that if the message has been signed with key K^{-1} , then A can prove that it has been signed with key K^{-1} without knowing K^{-1} . This is a reasonable assumption, since the key K , which

¹⁴The notion of belief is described in [2], [4].

is the only information that is required to verify the signature, is publicly known.

This postulate is used in proving that principals are accountable for the statements they sign. Using this postulate, a principal A can prove that another principal B is accountable for a statement which B has signed, if A can present B 's key certificate (which is issued by some trusted authority), and hence, prove that B 's signature can be authenticated by K .

By the definition of the `CanProve` construct, this postulate does not apply to a field x within a signed message that is encrypted with a shared key (unless the decrypted value of x is also a field in the signed message) since establishing that the signer is accountable for the plaintext counterpart of the encrypted field would require that the prover reveal a secret key. Since this is not a desirable way of proving accountability, in this paper (i.e., in the definition of `CanProve` construct), we require that, to prove accountability, the *prover* not reveal any secrets (that are different from the statement being proved) to the audience of a proof.

- *Trust Relationships*¹⁵

In systems using digital signatures (also in the paper world), often, proof of a statement x amounts to showing that the statement x has been endorsed by a (trusted) authority of x . For instance, in order to prove the validity of electronic currency, the prover (i.e., the payer) may be required to present a statement issued by a currency server endorsing the currency, to the recipient of the currency [19]. The following postulate uses this reasoning :

$$\text{Trust: } \frac{A \text{ CanProve } (B \text{ Says } x) ; \quad A \text{ CanProve } (B \text{ IsTrustedOn } x)}{A \text{ CanProve } x}$$

That is, if A can prove that B , who is an authority on x , says x , then A can prove that x holds. This postulate is based on two premises, namely, (1) when a trusted authority states x , it can prove x if needed to, and (2) a principal can prove x if it can prove that another principal can prove x .

VII. TRANSACTION GOALS

The accountability requirements are specific to each transaction. The examples of analysis goals presented here are neither necessary nor sufficient, but can be considered to be guidelines that are derived from the observation of rules that dictate the corresponding transactions in the paper world.

In a transaction that requires full accountability, when a *payor* makes a payment to a *payee*, the transaction goals may be, for instance,

$$\begin{aligned} \text{payor CanProve } (\text{payee received payment}) \dots(1) \\ \text{payee CanProve } (\text{payor made payment}) \dots\dots\dots(2) \end{aligned}$$

$$\text{payee CanProve } (\text{payment is valid}) \dots\dots\dots(3)$$

Here, the statements (*payee received payment*), (*payor made payment*) and (*payment is valid*) are interpretations of (digitally signed) protocol messages that are intended for receipt, payor identification and validation respectively.

The above generic transaction goals can be applied to specific transactions, such as *deposits*, *withdrawals*, and *peer payments*. In protocols for anonymous payments, the goal (2) above does not apply, but goals (1) and (3) do. In Section 8(D), using informal reasoning, we show that a recently published payment protocol [19] achieves only goal (3) above.

Further, in protocols for obtaining insured services, where a service token is accompanied by an endorsement of an official (e.g., [16]), the goals may be, for instance,

$$\begin{aligned} \text{service provider CanProve } (\text{service is insured}) \\ \text{service user CanProve } (\text{service is insured}) \end{aligned}$$

where the statement (*service is insured*) is an interpretation of a (digitally signed) message which is intended for insuring the service.

Other transactions, such as bids for proposals, contractual agreements (e.g., [26]) and so forth, may have transaction specific statements and goals of their own (see Section 8(A) for an example of some transaction goals). Most transaction goals of the commercial and legal world also apply to the electronic world. The message interpretations and transaction goals presented in this paper can be used as examples for interpreting the messages of, and formulating the goals of, other electronic transactions.

VIII. ANALYSIS EXAMPLES

This section shows the use of the proposed analysis framework in the analysis of protocols for their ability to establish accountability. The analysis consists of the following steps :

- stating a protocol's accountability goals (in protocols for electronic commerce, this is derived from the transaction goals),
- interpreting protocol messages,
- articulating initial state assumptions,
- analyzing messages for accountability properties, and
- comparing the provability results that protocol participants obtain, with the protocol goals. Failure of a protocol to achieve its goals indicates a weakness in the protocol which can be exploited.

Four protocol examples are chosen in this paper to illustrate the application of the postulates presented. In the first example, Carnegie Mellon's Internet Billing Server Protocol [23] alternative which uses only asymmetric encryptions is analyzed using the postulates, to show that it can achieve its goals using some assumptions and steps which are not required if minor modifications are made to the original protocol. The analysis also points to a redundancy in this protocol, which can be eliminated to make this protocol more optimal.

¹⁵I am grateful to Paul Syverson and Ross Anderson for providing useful insights on trust relationships.

In the second example, an “optimized” version of the IBS protocol that uses both symmetric and asymmetric encryptions is analyzed to show that it lacks many desirable accountability properties. Using the postulates, design modifications are suggested to remedy the protocol’s lack of accountability, and to remove its dependency on the assumption that the Billing Server is trusted.

In the third example, using the postulates, a recently published anonymous payment protocol [19] is analyzed for its properties to establish that this protocol does not achieve accountability properties, in agreement with the goals of this protocol.

The fourth example shows how the proposed postulates can be used to analyze protocols for delegation and public key distribution, by analyzing the SPX protocol [28]. This protocol is shown to achieve its desirable goals.

A. Internet Billing Server Protocol

Carnegie Mellon University’s Internet Billing Server (IBS) Protocol [23] is intended for offering billing services for service providers on the Internet. The major goals of this protocol are to ensure that the user cannot be billed for services that he is not rendered by the service provider at a previously agreed price, and the user cannot deny the charges made on his account by the service provider for the services rendered at a previously agreed price. The IBS protocol has three main components, price assurance, service provision and invoice delivery¹⁶. The three main steps, and their goals, are articulated below :

- *Price Assurance*

This is an agreement between the end-user (E) and the service provider (S) for obtaining a service at a mutually agreeable price. To have full accountability, the price assurance agreement should meet the following goals :

G1: $E \text{ CanProve } (S \text{ agreed on } (\frac{\text{price}}{\text{item}}))$

G2: $S \text{ CanProve } (E \text{ agreed on } (\frac{\text{price}}{\text{item}}))$

where $item$ is the unit with which the service is quantified (e.g., number of hours or number of database queries). The initial agreement between the end user and service provider can also include other terms and conditions for the service session.

- *Service Provision*

This is the session in which the service provider provides service to the end-user. Full accountability in this session would require that the following goals are met :

G3: $E \text{ CanProve } (S \text{ rendered } k \text{ items of service})$

G4: $S \text{ CanProve } (E \text{ received } k \text{ items of service})$

These goals require that E (S) has a way to establish that S rendered (E received) exactly k items of service.

¹⁶CMU-IBS protocol also has an initial step called Spending Cap agreement which we will not discuss, as it is not relevant for this analysis.

In the implementation, this requires that there be *start* and *end* markers for both service and acknowledgement messages, and that all intermediate messages have an associated serial number, so that the audience of the proof knows that the prover is presenting them with all (and only) the service messages.

- *Invoice Delivery*

This is a statement that the Billing Server (B) sends to the end user and the service provider informing them of the charges that were debited from the end users account and credited to the service providers account. To have full accountability, this protocol should meet the following goals :

G5: $E \text{ CanProve } (\$X \text{ transferred from } E \text{ to } S)$

G6: $S \text{ CanProve } (\$X \text{ transferred from } E \text{ to } S)$

Together, the price assurance goals and the service provision goals provide the end user and the service provider the ability to independently confirm or refute the invoice sent to them by the billing server if they are required to do so by a third party, such as a court of law. That is, from **G1** and **G3**, using the **Conj** and **Inf** postulates,

$$E \text{ CanProve } (E \text{ owes } S \ \$ (k \times \frac{\text{price}}{\text{item}}))$$

If **G5** is satisfied, and if X is different from $(k \times \frac{\text{price}}{\text{item}})$, then using the **Conj** and **Inf** postulates,

$$E \text{ CanProve } (X \text{ is invalid})$$

Further, from **G2** and **G4**, using the **Conj** and **Inf** postulates,

$$S \text{ CanProve } (E \text{ owes } S \ \$ (k \times \frac{\text{price}}{\text{item}}))$$

If **G6** is satisfied, and if X is different from $(k \times \frac{\text{price}}{\text{item}})$, then using the **Conj** and **Inf** postulates,

$$S \text{ CanProve } (X \text{ is invalid})$$

In [23], three protocol alternatives¹⁷ of the Internet Billing Server protocol are described, namely,

1. Protocol using symmetric keys only,
2. Protocol using asymmetric keys only, and
3. Protocol using a combination of symmetric and asymmetric keys.

It is observed in [23] that although the purely symmetric key based protocols are computationally the most inexpensive, they lack accountability assurances. The purely asymmetric key based protocol options on the other hand, offer accountability properties, but are computationally expensive. An optimal solution has been proposed in [23] by using both symmetric and asymmetric encryption in the

¹⁷For each of these protocol alternatives, two operational modes, batch and one-of, are described in [23]. For the purpose of this analysis, only the batch transactions are considered. The analyses of one-of transactions are similar.

same protocol. However, as shown in this analysis, this optimization causes a lack of accountability in the protocol.

Since the *symmetric key only* option is known to lack accountability, in this paper, we analyze only the *asymmetric key only* and the *symmetric and asymmetric key* (optimal) alternatives, in the following sections.

In the protocol descriptions below, the notation “ i) $A \rightarrow B : X$ ” is used to denote that message X is sent from principal A to B in the i^{th} message of the protocol. The notation $\{M\}_{K_a}$ is used to denote a message M encrypted (for confidentiality) with A 's public key K_a , and $\{M\}_{K_a^{-1}}$ is used to denote a message which is signed with A 's private key, where K_a and K_a^{-1} are the public and private counterparts of an asymmetric key pair. The notation $\{M\}_{K_{ab}}$ is used to denote a message M encrypted with the symmetric key shared between principals A and B .

In the analysis of protocols, unsigned messages do not offer any accountability, and hence, are not interpreted. While it may be possible to extend proof systems to analyze unsigned encrypted messages and/or plaintext messages, such extensions are beyond the scope of this paper.

B. Asymmetric Key IBS Protocol

In this protocol alternative of CMU's IBS [23] only asymmetric keys are used. The parties in the protocol are the end user (E), the service provider (S) and the billing server (B). This protocol is analyzed to show that it achieves its goals, but uses some additional steps and assumptions which can be avoided if some design modifications are made to the original protocol. The following analysis also indicates a redundancy in the protocol which can be eliminated to make this protocol more efficient.

As mentioned in Section 8(A), this protocol has three stages - *pricing agreement* between the end user and the service provider, *service provision* by the service provider to the end user, and the *invoice delivery* from the billing server to both service provider and end user. The message exchanges of the three protocol stages are listed and explained below.

Pricing Agreement:

- 1) $E \rightarrow S : \{ \textit{Price Request} \}_{K_e^{-1}}$
- 2) $S \rightarrow E : \{ \textit{Price} \}_{K_s^{-1}}$

In the pricing agreement protocol, the end user sends a pricing request to the service provider (S) in the first message, signed with the end user's private key (K_e^{-1}). The service provider responds with a price quote in a message signed with its private key K_s^{-1} . Here, the message field *Price* should be interpreted as $\frac{\textit{price}}{\textit{item}}$, where *item* is the unit with which the service is quantified.

Service Provision:

- 3) $E \rightarrow S : \{ \{ \textit{Price} \}_{K_s^{-1}}, \textit{Price} \}_{K_e^{-1}}$
- 4) $S \rightarrow \textit{Invoice} : \{ \{ \textit{Price} \}_{K_s^{-1}}, \textit{Price} \}_{K_e^{-1}}$
- 5) $S \rightarrow E : \{ \textit{Service} \}_{K_s^{-1}}$
- 6) $E \rightarrow S : \{ \textit{Service Acknowledge} \}_{K_e^{-1}}$

- 7) $S \rightarrow \textit{Invoice} : \{ \{ \textit{Service Acknowledge} \}_{K_e^{-1}} \}_{K_s^{-1}}$

In the service provision protocol, the user sends a service request to the service provider in the first message. The service provider copies this message to an invoice, and sends a signed service message to the user. The end user sends a signed service acknowledge, which the service provider signs and copies to its invoice.

Invoice Delivery:

- 8) $E \rightarrow S : \{ \textit{Invoice Request} \}_{K_e^{-1}}$
- 9) $S \rightarrow B : \{ \{ \textit{Invoice} \}_{K_b} \}_{K_s^{-1}}$
- 10) $B \rightarrow S : \{ \{ \textit{Invoice} \}_{K_s} \}_{K_b^{-1}}, \{ \{ \textit{Invoice} \}_{K_e} \}_{K_b^{-1}}$
- 11) $S \rightarrow E : \{ \{ \textit{Invoice} \}_{K_e} \}_{K_b^{-1}}$

In the invoice delivery protocol, the end user sends a signed invoice request to the service provider. The service provider sends its invoice to the billing server (B) in a message which is encrypted with the billing server's public key and then signed with the service provider's private key.

The billing server verifies the invoice and sends one invoice each to the service provider and end user, in separate encrypted and signed messages. In the last message, the service provider forwards the encrypted invoice to the end user.

Protocol Interpretation :

For the sake of brevity, only those messages which are relevant to the derivation of the protocol goals are interpreted below.

- 2) E Receives (*Price*) SignedWith K_s^{-1}
- 3) S Receives ((*Price*) SignedWith K_s^{-1} ,
Price) SignedWith K_e^{-1}
- 5) E Receives (*Service*) SignedWith K_s^{-1}
- 6) S Receives (*Service Acknowledge*) SignedWith K_e^{-1}
- 10) S Receives (*Encrypted Invoice*) SignedWith K_b^{-1} ,
(*Encrypted Invoice*) SignedWith K_b^{-1}
- 11) E Receives (*Encrypted Invoice*) SignedWith K_b^{-1}

Initial State Assumptions :

The initial state assumptions that are required in the analysis are listed below :

- A1:** S CanProve (K_e Authenticates E)
- A2:** E CanProve (K_s Authenticates S)
- A3:** S, E CanProve (K_b Authenticates B)
- A4:** (S Says *Price*) \Rightarrow (S agrees to ($\frac{\textit{price}}{\textit{item}}$))
- A5:** (E Says *Price*) \Rightarrow (E agrees to ($\frac{\textit{price}}{\textit{item}}$))
- A6:** (S Says *Service*) \Rightarrow (S renders one service item)
- A7:** (E Says *Service Acknowledge*) \Rightarrow
(E received one service item)

The first assumption states that the service provider can prove that the end user can be authenticated by the key K_e . That is, the service provider can prove that E is accountable for any message signed with K_e^{-1} . This assumption is

justified if either K_e is known to authenticate E globally, or if S can acquire and present a key certificate (from a certificate server or from a Directory) for the key K_e which is issued by a trusted authority. Again, if this authority is not globally trusted, additional key certificates and authorization certificates can be acquired from authorities who are higher up in the trust hierarchy such that the “certificate chain” is sufficient to prove that K_e **Authenticates** E (see Section 9 for more details on certificate chains).

The second assumption states that the end user can prove that the service provider can be authenticated by the key K_s . The third assumption states that the service provider and the end user can prove that the billing server can be authenticated with key K_b . The last four assumptions are about implications of what S and E state in messages 2, 3, 5 and 6, respectively.

Analysis :

Message 2:

When E receives message 2, using the assumption that E can prove the association of the signature on message 2 with S , (i.e., assumption **A2**), and applying the **Sign** postulate,

$$E \text{ CanProve } (S \text{ Says Price})$$

This statement can be further refined using assumption **A4** and the **Inf** postulate, as :

$$E \text{ CanProve } (S \text{ agrees to } (\frac{\text{price}}{\text{item}})) \dots\dots [G1]$$

Message 3:

When S receives message 3, using the assumption that S can prove the association of the signature on message 3 with E , (i.e., assumption **A1**), and applying the **Sign** postulate,

$$S \text{ CanProve } (E \text{ Says Price})$$

This statement can be further refined using assumption **A5** and the **Inf** postulate, as :

$$S \text{ CanProve } (E \text{ agrees to } (\frac{\text{price}}{\text{item}})) \dots\dots [G2]$$

Message 5:

When E receives message 5, using the same assumption as in message 2, and applying the **Sign** postulate,

$$E \text{ CanProve } (S \text{ Says Service})$$

Using assumption **A6**, and the **Inf** postulate, this is refined as

$$E \text{ CanProve } (S \text{ renders one service item})$$

If E has k such messages (or if the service message implies that k units of service are rendered), by using **Conj** and **Inf** postulates,

$$E \text{ CanProve } (S \text{ rendered } k \text{ Service items}) \dots [G3]$$

Message 6:

When S receives message 6, using the same assumption as that in message 3, and applying the **Sign** postulate,

$$S \text{ CanProve } (E \text{ Says Service Acknowledge})$$

Using assumption **A7**, and the **Inf** postulate, this is refined as

$$S \text{ CanProve } (E \text{ received one service item})$$

If S has k such acknowledgement messages (or if the service message implies that k units of service are rendered), by using **Conj** and **Inf** postulates,

$$S \text{ CanProve } (E \text{ obtained } k \text{ service items}) \dots\dots [G4]$$

Hence, this protocol achieves the desirable accountability goals in the *pricing agreement* and *service provision* stages. Using these results, both service user and service provider can independently prove the amount that should be debited from the user account and credited to the service provider’s account.

Further, this protocol permits the user and the service provider to prove to a third party that the invoice sent to them by the billing server is valid or invalid with some additional steps and assumptions which can be avoided with minor design modifications. The following analysis illustrates this point.

Message 10 :

When S receives messages 10, S can prove that B is accountable for sending the encrypted invoice (i.e., $\{Invoice\}_{K_s}$). However, in order to derive the statement $S \text{ CanProve } (B \text{ Says Invoice})$, S would have to send the encrypted message (i.e., $\{\{Invoice\}_{K_s}\}_{K_b^{-1}}$) as well as the plaintext *Invoice* to the audience of the proof.

The audience of this proof would then have to encrypt the plaintext *Invoice* with the public key of S , and verify that it matches the ciphertext in the message signed by B . Further, the audience would have to assume that B is the principal who encrypted the invoice with K_s , since it is possible for another principal (such as a trojan horse) to have encrypted the invoice and B to have signed the invoice without knowing its contents.

Hence, this protocol requires additional assumptions and steps for S to be able to hold B accountable for the plaintext value of the invoice (i.e., goal **G6**). The same holds for E ’s ability to hold B accountable for the invoice in message 11 (i.e., goal **G5**). However, a minor modification to this protocol allows the service provider as well as the user to hold the billing server accountable for his invoice, without going through the additional steps. The modified Invoice Delivery Protocol is :

- 8) $E \rightarrow S : \{ Invoice \ Request \}_{K_e^{-1}}$
- 9) $S \rightarrow B : \{ \{ Invoice \}_{K_s^{-1}} \}_{K_b}$

- 10) $B \rightarrow S : \{ \{Invoice\}_{K_b^{-1}} \}_{K_s}, \{ \{Invoice\}_{K_b^{-1}} \}_{K_e}$
 11) $S \rightarrow E : \{ \{Invoice\}_{K_b^{-1}} \}_{K_e}$

In this protocol, the invoices are signed first and then encrypted. Consequently, although the invoices are confidential, the recipients can hold the senders accountable for the invoices. That is, the following results are derivable :

$$\begin{aligned} E \text{ CanProve } (B \text{ Says } Invoice) & \dots\dots\dots [G5] \\ S \text{ CanProve } (B \text{ Says } Invoice) & \dots\dots\dots [G6] \end{aligned}$$

It is noteworthy that the derivation of accountability results did not require the interpretation of the first message of the *pricing agreement* protocol. Hence, this protocol can be optimized by sending this message un-signed (with its integrity protected by a checksum). However, such an optimization may introduce the possibility of the service provider sending price quotes to frivolous (un-signed) price requests.

This example shows that informal reasoning using the postulates can identify protocol weaknesses and protocol dependencies on assumptions. The analysis framework is used to suggest modifications in the design to strengthen the protocols.

C. Symmetric and Asymmetric Key IBS Protocol

In this protocol alternative of CMU's IBS [23] both asymmetric and symmetric keys are used. The parties in the protocol are the end user (E), the service provider (S) and the billing server (B). Although the resulting protocol alternative is more efficient than the first, it lacks some desirable accountability properties, as will be shown in the analysis below.

This protocol alternative makes the assumption that the billing server is trusted. We argue that this assumption is strong, since it implies that the end user must accept the invoices that he receives from the billing server, and hence, a compromise of the billing server can have unwarranted consequences¹⁸. Hence, this assumption indicates a weakness in the protocol which the financial institution owning the IBS can misuse.

Further, the following analysis shows that the dependency on the assumption of trust can be eliminated if the protocol is modified to achieve its desired accountability properties.

The message exchanges in the three protocol stages are described below :

Pricing Agreement:

- 1) $E \rightarrow S : \{Price\}_{K_{es}}$
 2) $S \rightarrow E : \{Price\}_{K_s^{-1}}$

In the pricing agreement protocol, in message 1, the end user sends the service provider a price request encrypted with the key K_{es} that it shares with the service provider. The service provider replies with a price quote signed with

¹⁸Although a user may trust his bank or credit card agency to manage his account, he verifies the monthly bills or statements he receives on his account, and can challenge the bank or credit card agency in court if there are discrepancies.

its private key, in message 2.

Service Provision:

- 3) $E \rightarrow S : \{ \{Price\}_{K_s^{-1}}, Price \}_{K_{eb}}, \{Price\}_{K_{es}}$
 4) $S \rightarrow Invoice : \{ \{Price\}_{K_s^{-1}}, Price \}_{K_{eb}}, \{Price\}_{K_s^{-1}}$
 5) $S \rightarrow E : \{Service\}_{K_s^{-1}}$
 6) $E \rightarrow S : \{Service\}_{K_{es}}, \{Service\}_{K_{eb}}$
 7) $S \rightarrow Invoice : \{ \{Service\}_{K_{eb}} \}_{K_s^{-1}}$

In the service provision protocol, the end user first sends the service provider a request for service in two message blocks, one encrypted with the key that it shares with the billing server (K_{eb}) and the other with the key it shares with service provider (K_{es}). Both message blocks contain the price information to which the end user has agreed. The first message block contains the price quote with the service provider's signature in it. The service provider copies this request and a signed message block containing the item and price information, to an invoice.

The service provider then sends a signed "Service" message to the end user. On receiving the service, the user sends an acknowledgement to the service provider encrypted with the key shared with the service provider, and a separate acknowledgement encrypted with the key shared with the billing server. The service provider copies the acknowledgement encrypted with the key shared by the end user and the billing server to the invoice, after affixing its signature on it.

Invoice Delivery:

- 8) $E \rightarrow S : \{Invoice\}_{K_{es}}$
 9) $S \rightarrow B : \{ \{Invoice\}_{K_b} \}_{K_s^{-1}}$
 10) $B \rightarrow S : \{ \{Invoice\}_{K_{eb}} \}_{K_b^{-1}}, \{ \{Invoice\}_{K_s} \}_{K_b^{-1}}$
 11) $S \rightarrow E : \{ \{Invoice\}_{K_{eb}} \}_{K_b^{-1}}$

In the invoice delivery protocol, the end user first sends the service provider an invoice request encrypted with the key K_{es} shared with the service provider. The service provider sends an invoice to the billing server encrypted with the billing server's public key and signed with the service provider's private key.

The billing server then sends the service provider two copies of the invoice, one encrypted with the key it shares with the end user and the other encrypted with the service provider's public key, and both signed with the billing server's private key. The service provider then forwards the user's copy of the encrypted invoice to the user.

Protocol Interpretation :

Only those messages which are signed, and have plaintext contents, are useful in this analysis, and hence only such messages are interpreted below.

- 2) E Receives ($Price$) SignedWith K_s^{-1}
 5) E Receives ($Service$) SignedWith K_s^{-1}

Initial State Assumptions :

The initial state assumptions that are required in the analysis are listed below :

- A1:** $S \text{ CanProve } (K_e \text{ Authenticates } E)$
- A2:** $E \text{ CanProve } (K_s \text{ Authenticates } S)$
- A3:** $S, E \text{ CanProve } (K_b \text{ Authenticates } B)$
- A4:** $(S \text{ Says } Price) \Rightarrow (S \text{ agrees to } (\frac{price}{item}))$
- A5:** $(S \text{ Says } Service) \Rightarrow (S \text{ renders one service item})$

The first assumption states that S can prove that E can be authenticated using key K_e . The second assumption states that E can prove that K_s authenticates S , and the third assumption states that both S and E can prove that K_b authenticates B , respectively. The fourth and fifth assumptions are about the interpretation of the statements that S signs in this protocol.

Analysis :

Message 2:

When E receives message 2, using the assumption that E can prove the association between the signature on message 2 with S (i.e., assumption **A2**), and using the **Sign** postulate,

$$E \text{ CanProve } (S \text{ Says } Price)$$

This can be refined using the assumption **A4** and the **Inf** postulate, as

$$E \text{ CanProve } (S \text{ agrees on } (\frac{price}{item})) \dots\dots [G1]$$

Message 5:

When E receives message 5, using the same assumptions as in message 2, and using the **Sign** postulate,

$$E \text{ CanProve } (S \text{ Says } Service)$$

Using assumption **A5**, and the **Inf** postulate, this is refined as

$$E \text{ CanProve } (S \text{ renders one service item})$$

If E has k such messages (or if the service message implies that k units of service are rendered), by using **Conj**,

$$E \text{ CanProve } (S \text{ rendered } k \text{ service items}) \dots\dots [G2]$$

The price agreement and service provision protocols do not provide any additional accountability results (i.e., do not achieve goals **G3** and **G4**) in this protocol alternative of IBS. Consequently, the service provider can neither prove that the user agreed on a price, nor that the user acknowledged the service, since both these messages are encrypted with the key that the service provider shares with the user (the service provider could just as easily generate these messages by itself).

Full accountability can be restored in the service session if the following minor modifications are performed on messages 3 and 6.

- 3) $E \rightarrow S : \{ \{ Price \}_{K_s^{-1}}, Price \}_{K_{eb}}, \{ Price \}_{K_e^{-1}}$
- 6) $E \rightarrow S : \{ Service \ Acknowledge \}_{K_e^{-1}}, \{ Service \ Acknowledge \}_{K_{eb}}$

With the modified protocol, the goals **G1** through **G4** can be achieved, at the cost of two additional signatures, and with the elimination of a strong assumption (i.e., that the billing server is trusted) and two shared key encryptions.

Further, the invoice delivery protocol of this IBS protocol alternative achieves goal **G6** with the same assumption and steps as in Section 8(B), but fails to achieve goal **G5**.

Message 10 :

The analysis of this message is identical to that of message 10 in the *asymmetric key only* protocol. S can hold B accountable for *Invoice* (i.e., achieve goal **G6**) using the additional steps and assumptions which are described in Section 8(B). The proposed modifications to this protocol (below) eliminates the need for these additional steps and assumptions.

Message 11:

When E receives message 11, E can prove that B is accountable for $\{ Invoice \}_{K_{eb}}$, but cannot prove that B is accountable for *Invoice* unless it reveals the secret key K_{eb} . Hence, this protocol does not achieve goal **G5**. Consequently, the end user cannot hold the billing server responsible for the invoice that he receives in this protocol.

In the original protocol, this lack of accountability may not be perceived as a problem, since the billing server is assumed to be trusted, and the role of the invoice is primarily that of informing the user and service provider of the service charges. However, a minor modification to this protocol allows the user to hold the billing server accountable for his invoice (i.e., goal **G5**), without requiring an assumption of trust on the billing server, and without requiring the additional steps and assumptions that were necessary in the original protocol to achieve goal **G6**. The modified Invoice Delivery Protocol would be :

- 8) $E \rightarrow S : \{ Invoice \ Request \}_{K_{es}}$
- 9) $S \rightarrow B : \{ \{ Invoice \}_{K_s^{-1}} \}_{K_{bs}}$
- 10) $B \rightarrow S : \{ \{ Invoice \}_{K_b^{-1}} \}_{K_{eb}}, \{ \{ Invoice \}_{K_b^{-1}} \}_{K_{bs}}$
- 11) $S \rightarrow E : \{ \{ Invoice \}_{K_b^{-1}} \}_{K_{eb}}$

In this protocol, the invoices are signed first and then encrypted. Consequently, although the invoices are confidential, the recipients can hold the senders accountable for the invoices signed. That is, the following results are derivable

$$E \text{ CanProve } (B \text{ Says } Invoice) \dots\dots\dots [G5]$$

$$S \text{ CanProve } (B \text{ Says } Invoice) \dots\dots\dots [G6]$$

This example shows that informal reasoning using the pos-

tulates can identify protocol weaknesses and protocol dependencies on assumptions. The analysis can be used to suggest modifications in the design to strengthen protocols requiring accountability.

D. USC ISI Payment Protocol

The following payment protocol, due to Medvinsky and Neuman [19] (Figure 1), is intended for payment from payor (A) to payee (B), where A remains anonymous. This protocol is analyzed to show that the postulates can point to a lack of accountability in this protocol [19]. It is noteworthy that the weakness pointed to by this analysis is a consequence of the protocol goals, and that it has been observed and remedied by the authors of [19].

Protocol Description:

In this protocol, initially, (i.e., in messages 1 and 2) the payor (A) obtains the payee's (B 's) key which the payee uses to keep its identity anonymous. In message 3, A sends the coins, the identifier of the desired service S_{id} , and two keys SK_{AN1} and K_{ses} encrypted with the key that B provides.

Using K_{CS} , B can verify that the coin is valid (i.e., that it has been issued by a certified currency server). The session key K_{ses} is used by B to identify A during service. After receiving the coins from A , B verifies the coins with the currency server in messages 4 and 5. If the coins haven't already been spent, the server issues new coins for B (message 5). In the last message (message 6), B returns a receipt signed with the private key (i.e., the private counterpart of the key sent in message 2) encrypted with SK_{AN1} for confidentiality. The receipt includes the amount paid, date, and a unique identifier T_{id} that will be used along with the session key to obtain the service.

The actual protocol description is given below using symbolic notation.

- 1) $A \rightarrow B : K_{AN}$
- 2) $B \rightarrow A : \{K_{BN}\}_{K_{AN}}$
- 3) $A \rightarrow B : \{ \{coins\}_{K_{CS}^{-1}}, SK_{AN1}, K_{ses}, S_{id} \}_{K_{BN}}$
- 4) $B \rightarrow CS : \{ \{coins\}_{K_{CS}^{-1}}, SK_{BN}, transaction \}_{K_{CS}}$
- 5) $CS \rightarrow B : \{ \{new_coins\}_{K_{CS}^{-1}} \}_{SK_{BN}}$
- 6) $B \rightarrow A : \{ \{amount, T_{id}, date\}_{K_{BN}^{-1}} \}_{SK_{AN1}}$

As the authors point out, the protocol is vulnerable because B could simply spend A 's coin without providing a valid receipt. However, the protocol has yet another vulnerability which is a direct consequence of B 's anonymity. That is, in spite of A receiving a receipt from B , A cannot hold B accountable for the service. The analysis of this protocol using the postulates makes this weakness (which has already been observed in [19]) explicit.

Protocol Interpretation :

Messages 1, 2 and 3 are not signed and hence are not interpreted. Message 4 is not interpreted since it is received by the currency server - this analysis only pertains to what the

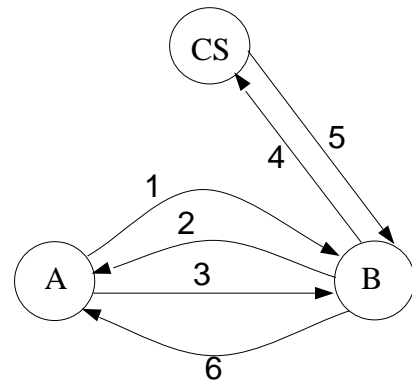


Fig. 1. The Medvinsky-Neuman Protocol

payor and the payee can prove at the end of the protocol.

- 5) B Receives (*coins valid*) SignedWith K_{CS}^{-1}
- 6) A Receives (*receipt of payment*) SignedWith K_{BN}^{-1}

Initial State Assumptions :

The initial state assumptions that are required in the analysis are listed below :

- A1:** A, B, CS CanProve (K_{CS} Authenticates CS)
- A2:** B CanProve (CS IsTrustedOn *coins_valid*)

The first assumption states that A, B and CS can prove that K_{CS} authenticates CS . The second assumption states that B can prove that CS is trusted on the statement on the coins being valid. That is, in this system, if a principal can prove that CS says that the coins are valid, then that principal has adequately proved that the coins are valid.

Analysis:

Message 5:

When B receives message 5, using the initial state assumption that B can prove that K_{CS} authenticates CS (assumption **A1**), and applying the **Sign** postulate, we have that

$$B \text{ CanProve } (CS \text{ Says } \textit{coins_valid})$$

The above result can be further refined using the initial state assumption that B can prove that CS is trusted on certifying the coins (i.e., assumption **A2**), and applying the **Trust** postulate. From this we obtain,

$$B \text{ CanProve } (\textit{coins_valid})$$

B cannot establish that the payment was made by A , in agreement with the goals of this protocol (i.e., to protect the payor's anonymity).

Message 6:

As the authors indicate, if A does not receive message 6, then A cannot prove that B received the payment. This allows a dishonest principal B to accept the payment without

sending a receipt, with the payor not being able to prove that he made the payment.

However, the protocol also has another, perhaps more subtle, vulnerability. In spite of A receiving message 6, it cannot use this information to hold B accountable for the service. The message interpretation is :

6) A **Receives** ((*receipt of payment*) SignedWith K_{BN}^{-1})

However, we cannot apply the **Sign** postulate on this message, since the key K_{BN}^{-1} is not known (or assumed) to authenticate any principal. To apply the **Sign** postulate, (to derive A **CanProve** B **Says** (*receipt of payment*)), we need the additional property that A can prove that K_{BN} authenticates B .

This protocol weakness is a direct consequence of the protocol option which allows B to send A a key of its choice (K_{BN}), and follows from the goals of this protocol (i.e., to maintain B 's anonymity) as stated by Medvinsky and Neuman. Accountability can be had in this protocol if either A is made responsible for retrieving B 's key certificate from the Directory Service, or if B presents this certificate to A in message 2. In the second solution, for instance, the first two messages of the modified protocol may be :

1) $A \rightarrow B : K_{AN}$
 2) $B \rightarrow A : \{ \{B's\ Key\ Certificate\}_{K_{CA}^{-1}} \}_{K_{AN}}$

where CA is an authority which is trusted to certify B 's key. In [19], the authors present another protocol that guarantees A a valid receipt.

This simple example shows that, since accountability and anonymity are conflicting goals, this protocol can be vulnerable if used for transactions that require accountability.

E. SPX Authentication Exchange (with support for Delegation)

In this example, we show the application of the postulates to the analysis of public key distribution protocols for delegation. In the past, delegation has been studied in [1] using a logical calculus. In this paper, delegation is studied from the perspective of the delegate's provability of the delegator's accountability.

The generic goals of public key distribution with delegation are,

Client CanProve (K_s **Authenticates** *Server*) [G1]
Server CanProve (K_c **Authenticates** *Client*) [G2]
Delegate CanProve (K_{Del} **Authenticates** *Delegator*) ... [G3]

In SPX [28], principals authenticate each other by exchanging *authentication tokens*. An authentication token securely transfers a session key generated by the claimant and encrypted with the verifier's public key. A simplified version of the SPX authentication exchange [28] is shown in the Figure 2 and is described below. A user principal, the claimant, requests from the Certificate Distribution Center (CDC) a certificate for a server S , who plays the role of the

verifier of the claimant's credentials. The claimant verifies this certificate using the public keys of its trusted certification authorities (CAs). It then sends a message containing its name, a ticket, including the delegation public key and lifetime signed with its long term private key, a random session key encrypted with the verifiers' public key and the delegation private key encrypted with the session key.

The verifier recovers the session key using its private key, and retrieves the delegation private key using this session key. It checks this key with the delegation public key that is in the delegation ticket. If they match, the verifier knows that it has a good authenticator, but not who made it (since it does not have the public key of the claimant yet). In the last two messages, the verifier retrieves the claimants' credentials and uses the public key of the claimant to verify his signature on the delegation ticket. The claimant's credentials are then installed using the delegation ticket and the delegation private key.

Protocol Description:

1) $C \rightarrow CDC : S$
 2) $CDC \rightarrow C : \{ \{S, K_s, TA_1\}_{K_{TA_1}^{-1}} \}_{K_{CDC}^{-1}}$
 3) $C \rightarrow S : \{K_{Del}, T\}_{K_c^{-1}}, \{K_{des}\}_{K_s}, \{K_{Del}^{-1}\}_{K_{des}}$
 4) $S \rightarrow CDC : C$
 5) $CDC \rightarrow S : \{ \{C, K_c, TA_2\}_{K_{TA_2}^{-1}} \}_{K_{CDC}^{-1}}$
 6) $S \rightarrow C : Response (accept/reject)$

In the above protocol description, C is the claimant, S is the verifier and CDC is the certificate distribution center. In the first message, C sends to CDC the identity of the verifier. In message 2, CDC sends C a certificate which is issued by a trusted authority, TA_1 . C now sends its delegation public key (note that this portion of the message is only signed), a secret DES key K_{des} (encrypted with S 's public key) and the delegation private key encrypted with this secret key.

Although S does not have K_c , S can obtain K_{Del} from message 3, since typically, signatures do not protect message content confidentiality. S can also obtain K_{des} by decrypting $\{K_{des}\}_{K_s}$, and use K_{des} to decrypt $\{K_{Del}^{-1}\}_{K_{des}}$. S can verify the validity of K_{Del} using K_{Del}^{-1} . However, since S does not have the key certificate to C 's public key, it cannot hold C accountable for the delegation public key.

In an attempt to verify the signature on the delegation key of C , S sends C 's identity to CDC and obtains a key certificate (which is issued by TA_2) from CDC . S uses the public key of C in this certificate to authenticate C 's signature on the delegation key. If S is convinced of C 's delegation key, then it responds in the last message with an 'accept'.

Principals CDC , TA_1 and TA_2 play the roles of trusted authorities who have the authority to make certain statements. The protocol messages are interpreted as follows :

Protocol Interpretation:

The statements (K_s **Authenticates** S), (K_{Del} **Authenticates**

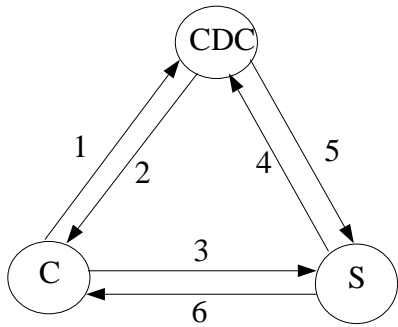


Fig. 2. The Tardo-Alagappan (SPX) Protocol for Delegation

C during T) and $(K_C \text{ Authenticates } C)$ convey the semantics of the message fields of messages 2, 3 and 5 which are relevant to this analysis. The protocol message interpretations are :

2) C Receives

$((K_S \text{ Authenticates } S) \text{ SignedWith } K_{TA_1}^{-1})$

5) S Receives $((K_C \text{ Authenticates } C) \text{ SignedWith } K_{TA_2}^{-1})$

$\text{SignedWith } K_{CDC}^{-1}$
 $\text{SignedWith } K_{CDC}^{-1}$

3) S Receives $(K_{Del} \text{ Authenticates } C \text{ during } T)$

$\text{SignedWith } K_C^{-1}$

The fifth message is analyzed before message 3 in this protocol. This is because the protocol design is such that the key K_C derived from message 5 is necessary for authenticating the signature in message 3.

Initial State Assumptions:

The initial state assumptions that are required in the analysis are listed below :

- A1:** $C, S \text{ CanProve } (K_{CDC} \text{ Authenticates } CDC)$
- A2:** $C \text{ CanProve } (K_{TA_1} \text{ Authenticates } TA_1)$
- A3:** $S \text{ CanProve } (K_{TA_2} \text{ Authenticates } TA_2)$
- A4:** $C, S \text{ CanProve } (CDC \text{ IsTrustedOn } (CDC \text{ Says}))$
- A5:** $C \text{ CanProve } (TA_1 \text{ IsTrustedOn } (TA_1 \text{ Says}))$
- A6:** $S \text{ CanProve } (C \text{ IsTrustedOn } (K \text{ Authenticates } C))$
- A7:** $S \text{ CanProve } (TA_2 \text{ IsTrustedOn } (TA_2 \text{ Says}))$

The assumptions are similar to those in the previous examples. Assumptions **A1**, **A2** and **A3** are about asymmetric keys. They state, in effect, that the association of principals to statements can be proved (using public key certificates).

Assumptions **A4**, **A5** and **A7** can be justified if either (1) the authorities CDC , TA_1 and TA_2 are globally trusted, or (2) principals C and S can obtain the authorization certificates (or certificate chains) that are necessary to establish trust on what principals CDC , TA_1 or TA_2 state.

Assumption **A6** states, in effect, that it can be proven that the principal C is trusted on announcing its own delegation key. Since C is liable for the messages signed with

the delegation private key, it is in its own interest to exercise judgement in making this announcement.

Analysis:

Message 2:

When C receives message 2, using the assumption that C can prove the association of CDC with CDC 's signature (i.e., assumption **A1**), and applying **Sign** postulate,

$C \text{ CanProve } (CDC \text{ Says } ((K_S \text{ Authenticates } S) \text{ SignedWith } K_{TA_1}^{-1}))$

Using assumption **A4** about the trust on CDC , and applying the **Trust** postulate on the result derived above, we have,

$C \text{ CanProve } ((K_S \text{ Authenticates } S) \text{ SignedWith } K_{TA_1}^{-1})$

Using the assumption that the C can prove that K_{TA_1} authenticates TA_1 (i.e., assumption **A2**), and applying the **Sign** postulate again, we have

$C \text{ CanProve } (TA_1 \text{ Says } (K_S \text{ Authenticates } S))$

Further, using assumption **A5** that C can prove that TA_1 is trusted to make the above statement, and applying the **Trust** postulate, we have

$C \text{ CanProve } (K_S \text{ Authenticates } S) \dots\dots\dots [G1]$

Hence, this protocol empowers C to prove that K_S authenticates S , hence, giving C the ability to hold the server responsible for any message signed with K_S .

Message 5:

For the sake of this analysis, it is useful to analyze message 5 before we analyze message 3. The analysis is similar to that of message 2, and will not be repeated. Application of the **Sign** postulate and the **Trust** postulate on the interpreted message, and the using assumptions **A1**, **A3**, **A4** and **A7**, the resulting statement is:

$S \text{ CanProve } (K_C \text{ Authenticates } C) \dots\dots[G2]$

Hence, this message empowers S to prove that K_C authenticates S . This allows S to prove that C is responsible for any statement signed with key K_C . This is useful in delegation, since message 3, which contains the delegation key K_{Del} , is signed by C (the delegator) using its private key K_C^{-1} .

Message 3:

The derivation is similar to that of message 2. However, in analyzing this message, we make use of the result obtained from the previous message, namely that S can prove that K_C authenticates C . Application of the **Sign** postulate and **Trust** postulate on the interpreted message 3, and using

assumption **A6**, the resulting statement is:

S CanProve (K_{Del} Authenticates C during T) [G3]

Hence, S can prove to a third party that it is acting on C 's behalf when it uses the key K_{Del} . This example illustrates the use of *provability* in delegation.

The results of this analysis show that this protocol achieves its goals, namely that of empowering C and S to be able to hold each other accountable for their statements made after the protocol, and for S (the delegate) to prove that the delegation key authenticates C (the delegator) during T .

These examples show that the proposed postulates are useful in deriving accountability properties of protocols for electronic commerce. The last example shows that the analysis is extensible to key distribution and delegation protocols. In what follows, the proof of accountability is shown to require chained endorsements in the absence of globally trusted authorities.

IX. ENDORSEMENT CHAINS

In asymmetric encryption systems, the proof of accountability of a message typically involves two components:

- Signed message, and
- Proof of a unique identity associated with signature.

If the signature on the message cannot be assumed to be associated with a unique principal, the second component above may, in turn, require a signed message endorsing this association. This may be in the form of a key certificate signed by a trusted authority. If key certifying authorities are not globally trusted, then a certificate chain may be required. For instance, the proofs of identity may be in the form of a key certificate chain, such as:

$$\begin{aligned} & \{K \text{ Authenticates } P_1\}_{KP_2^{-1}} \\ & \{K P_2 \text{ Authenticates } P_2\}_{KP_3^{-1}} \\ & \dots \\ & \{K P_{n-1} \text{ Authenticates } P_{n-1}\}_{KP_n^{-1}} \end{aligned}$$

where P_1, P_2, \dots, P_n are key certifying authorities, and KP_n^{-1} is known to authenticate P_n by the audience of the proof of accountability of P_1 . If the key certifying authorities are not trusted to certify the keys of other principals by the audience of the proof, in order to prove accountability, in addition to the above key certificate chain, an authorization chain may be required, such as :

$$\begin{aligned} & \{P_2 \text{ IsTrustedOn } (K \text{ Authenticates } P_1)\}_{KA_2^{-1}} \\ & \{P_3 \text{ IsTrustedOn } (K P_2 \text{ Authenticates } P_2)\}_{KA_3^{-1}} \\ & \dots \\ & \{P_k \text{ IsTrustedOn } (K P_{n-1} \text{ Authenticates } P_{n-1})\}_{KA_k^{-1}} \end{aligned}$$

where A_2, A_3, \dots, A_k are trusted authorities. Additional certificates to certify the signatures and the authorizations of these authorities may be necessary if they cannot be assumed to be trusted by the audience of the proof.

Ideally, the key certificate chains should terminate in certificates issued by principals whose identity association with their signature is known by the audience of a proof, such that the accountability of all statements that are required to establish accountability of P_1 can be proved. Authorization certificate chains terminate in principals who are trusted to authorize other principals, such that all statements that are required to establish the accountability of P_1 can be considered authoritative.

X. CONCLUSIONS

Accountability is an important requirement in most electronic commerce transactions. In designing protocols for electronic commerce, it is important to identify the protocol messages that must provide accountability assurances, for at least two reasons, namely, (1) the lack of adequate accountability properties can make these protocols susceptible to disputes between transacting parties due to insufficient evidence, and (2) redundant accountability properties can make them inefficient, since accountability is typically provided using the computationally expensive asymmetric encryption. Articulating the transaction goals and analyzing the protocol messages as shown in this paper provides a systematic approach for determining the optimal set of protocol messages which should provide accountability assurances, and the initial state assumptions that are required to achieve the desired goals.

The examples chosen in this paper illustrate the application of the proposed analysis framework. The first three examples show that the framework is useful in detecting and correcting lack of accountability in protocols, or the dependency of protocols on strong assumptions to achieve desirable accountability goals. The last example shows that the provability postulates can be applied to the analysis of not only electronic commerce protocols, but to any protocol in which accountability is important.

Analysis of this type can guide protocol designers and analysts to make explicit message interpretations, protocol goals and initial state assumptions, and consequently, to examine the adequacy of the protocols for their desired goals, or the validity of the assumptions that they require.

The framework for analysis proposed does not attempt to solve all problems with protocols for electronic commerce. The reasoning in this paper focuses only on accountability properties of protocols. Consequently, the analysis can only detect protocol weaknesses due to lack of accountability, which may appear in the absence of, or in addition to, other problems, such as lack of freshness. The analysis may not be applicable to protocols in which accountability is not required or desired.

The application of this framework for the analysis of proofs in general, and the analysis of accountability in other types of systems, are areas that are open to future research.

ACKNOWLEDGEMENTS

Whatever clarity this paper has is due in no small measure to comments received from Paul Syverson, Ross Anderson, Clifford Neuman, Roger Needham, Martin Abadi

and Catherine Meadows. I would also like to thank Li Gong, Marvin Sirbu, Benjamin Cox, Charlie Watt, Genady Medvinsky, Robert Fourney and the anonymous referees for the numerous useful comments on earlier versions of this paper.

REFERENCES

- [1] M.Abadi, M.Burrows, B.Lampson and G.Plotkin, "A Calculus for Access Control in Distributed Systems", *Proceedings of Crypto'91*, Springer Verlag 1992. Also research report 70, Systems Research Center, Digital Equipment Corporation, Palo Alto, March 1991.
- [2] M.Abadi and M.Tuttle, "A semantics for a logic of authentication", *Proceedings of the 10th Annual ACM Symposium on Principles of Distributed Computing*, August 1991.
- [3] P.W.Brown, "Digital Signatures: Can they be accepted as Legal Signatures in EDI?", In *Proceedings of the ACM Conference on Computer and Communication Security*, November 1993.
- [4] M.Burrows, M.Abadi and R.Needham, "A Logic of Authentication", *ACM Transactions on Computer Systems*, 8(1), February 1990.
- [5] D.Chaum, A.Fiat, and N.Naor, "Untraceable electronic cash", In *Proceedings of Crypto '88*, 1988.
- [6] D.Chaum, H. van Antwerpen, "Undeniable Signatures", *Advances in Cryptology - Proceedings of CRYPTO '88*, Berlin: Springer-Verlag, 1990.
- [7] D.Chaum, B.Boer, E.Heyst, S.Mjolsnes and A.Steenbeek. "Efficient off-line electronic checks", In *Proceedings of Eurocrypt '89*, 1989.
- [8] R.A.De Millo, R.J.Lipton and A.J.Perlis, "Social Processes and Proofs of Theorems and Programs", In *Communications of the ACM*, Volume 22, Number 5, May 1979.
- [9] W.Diffie and M.E.Hellman, "New Directions in Cryptography", *IEEE Transactions on Information Theory*, Vol. IT-22, No. 6, 1976.
- [10] V.D.Gligor, R.Kailar, S.Stubblebine, L.Gong, "Logics for Cryptographic Protocols - Virtues and Limitations", *IEEE Computer Security Foundations Workshop*, June 1991.
- [11] L.Gong, R.Needham, and R.Yahalom, "Reasoning about Belief in Cryptographic Protocols", in *Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy*, Washington (IEEE), 1990.
- [12] L.C.Guillou, J.-J.Quisquater, "A 'Paradoxical' Identity-Based Signature Scheme Resulting from Zero-Knowledge," *Advances in Cryptology - Proceedings of CRYPTO '88*, Berlin: Springer-Verlag, 1990.
- [13] R.Kailar, "Reasoning about Accountability in Protocols for Electronic Commerce", In *Proceedings of the IEEE Symposium on Security and Privacy*, May 1995.
- [14] R.Kailar, V.Gligor, "On Belief Evolution in Authentication Protocols", In *Proceedings of the IEEE Computer Security Foundations Workshop*, June 1991.
- [15] R.Kailar, V.Gligor, L.Gong, "Effectiveness Analysis of Cryptographic Protocols", In *Proceedings of the IFIP Conference on Dependable Computing for Critical Applications*, January 1995.
- [16] C.Lai, G. Medvinsky, B.C.Neuman, "Endorsements, Licensing, and Insurance for Distributed System Services", In *Proceedings of the ACM Conference on Computer and Communication Security*, November 1994.
- [17] Y.I.Manin. *A Course in Mathematical Logic*. Springer-Verlag, 1977.
- [18] C. Meadows, "Using Narrowing in the Analysis of Key Management Protocols", In *Proceedings of the IEEE Computer Symposium on Security and Privacy*, May 1989.
- [19] G. Medvinsky, B.C.Neuman, "NetCash: A design for practical electronic currency on the Internet", In *Proceedings of the ACM Conference on Computer and Communication Security*, November 1993.
- [20] J.K.Millen, S.C.Clark, S.B.Freedman, "The Interrogator: Protocol Security Analysis", In *IEEE Transactions on Software Engineering*, Vol. 13, No. 2, March 1987.
- [21] B.C. Neuman, G. Medvinsky, "Requirements for Network Payment: The NetChequeTM Perspective", In *Proceedings of the IEEE CompCon'95*, San Francisco, March 1995.
- [22] T.Okamoto and K.Ohta, "Universal electronic cash", In *Proceedings of Crypto '91*, 1991.
- [23] K.R.O'Toole, "The Internet Billing Server Transaction Protocol Alternatives", Carnegie Mellon University Information Networking Institute, INI TR 1994-1, April 1994.
- [24] B.Pfitzmann and M.Waidner, "How to break and repair a 'provably secure' untraceable payment system", In *Proceedings of Crypto '91*, 1991.
- [25] R.Anderson, "Why Cryptosystems Fail", In *Proceedings of the ACM Conference on Computer and Communication Security*, November 1993.
- [26] B.Schneier, "Applied Cryptography", *John Wiley and Sons, Inc.*, 1993.
- [27] P.Syverson, "The Use of Logic in the Analysis of Cryptographic Protocols", In *Proceedings of IEEE Symposium on Research in Security and Privacy*, May 1991.
- [28] J.Tardo and K.Alagappan, "SPX: Global authentication using public key certificates", in *Proceedings of the IEEE Symposium on Security and Privacy*, May 1991.
- [29] V.L.Voydock, S.T.Kent, "Security Mechanisms in High-level Network Protocols", in *Computing Surveys*, Vol. 15, No. 2, 1983.
- [30] C.Wilder, "Digital Dollars", *Information Week*, October 31, 1994.