Verifying the SET Purchase Protocols

Giampaolo Bella
Univ. of Cambridge

Fabio Massacci
Univ. of Trento

Lawrence C. Paulson
Univ. of Cambridge

November 2001

Abstract

The Secure Electronic Transaction (SET) protocol has been proposed by a consortium of credit card companies and software corporations to guarantee the authenticity of e-commerce transactions and the confidentiality of data. When the customer makes a purchase, the SET dual signature keeps his account details secret from the merchant and his choice of goods secret from the bank.

This paper reports verification results for the purchase step of SET, using the inductive method. The credit card details do remain confidential. The customer, merchant and bank can confirm most details of a transaction even when some of those details are kept from them. The usage of dual signatures requires repetition in protocol messages, making proofs more difficult but still feasible.

The formal analysis has revealed a significant defect. The dual signature lacks explicitness, giving rise to potential vulnerabilities.

Contents

1 Introduction

SET (Secure Electronic Transaction) is a huge suite of protocols devised by Visa and Mastercard for on-line shopping. In this paper, we focus on the *purchase phase* of SET and its key construct: the *dual signature*. This mechanism lets the customer agree the order details with the merchant while hiding those details from the bank; at the same time, it lets the customer share his credit card details with the bank while hiding them from the merchant.

We are looking at SET because within it are ideas of scientific interest for security verification and security protocol design. For example, PKCS digital envelopes [17] will be used in future protocols whether or not SET is a commercial success. Moreover, its sheer size, over 1000 pages of official documentation [8, 9, 10, 11], makes it a challenge for formal verification.

We have carefully simplified SET to make its analysis tractable, but we have retained the most important mechanisms. Our simplified version of SET is still one of the most complex protocols ever to be analysed formally.

We have found that, on the whole, dual signatures seem to work: the credit card details do remain confidential and still the various parties can be sure that they are dealing with the same transaction, even if they possess only partial information. Unfortunately, the dual signature, in common with many other SET messages, violates the explicitness principle of Abadi and Needham [1]. So some guarantees are weaker than they should be — particularly for the Payment Gateway, whose function is to authorize transactions.

In other papers [3], we have described modelling issues of the general SET protocol. The present paper describes work on verifying SET's *Purchase* phases using the inductive method and the Isabelle theorem prover.

Paper outline. In the next sections we present an overview of SET (§2) and of its purchase phase (§3). We discuss the formal model, presenting the most complicated rule using Isabelle syntax (§4). Then we discuss the proofs (§5) and conclude with a brief discussion of related work (§6).

2 SET Overview

Most Internet merchants use the SSL protocol to prevent eavesdroppers from learning customers' account details, such as credit card numbers. This arrangement follows the classical idea that bad persons are necessarily outsiders, and it has two major limitations:

- The customer has to trust the merchant to keep these details secure. Some merchants are dishonest or at best incompetent. A million credit

card numbers have recently been stolen from Internet sites whose managers had not applied security patches [13].

- The merchant has to trust the customer, who does not sign anything. The merchant has little protection from the use of stolen card numbers or from customers who repudiate their purchases.

Visa and Mastercard designed the SET protocol to address this unsatisfactory situation. Specifically, it should [9, page 6]

1. Provide confidentiality of payment information
2. Ensure integrity of all transmitted data
3. Provide authentication that a cardholder is a legitimate user of a branded payment card account
4. Provide authentication that a merchant can accept branded payment card transactions

To achieve these goals, the SET protocol comprises five main sub-protocols:

- *Cardholder Registration* allows a customer to register a credit card with a Certificate Authority. The request includes the Cardholder's public signature key and a secret nonce. The outcome of registration is a public-key certificate that includes the hash of the *primary account number* (PAN), i.e. the credit card number, and of a secret nonce (PANSecret).
- *Merchant Registration* is analogous. A Merchant registers both a signature key and an encryption key.
- *Purchase Request* allows a Cardholder to place an order with a Merchant.
- *Payment Authorization* follows or is combined with *Purchase Request*. It allows a Merchant to verify the Cardholder's details with a so-called Payment Gateway, which authorizes the transactions.
- *Payment Capture* allows a Merchant to request the actual transfer of funds.

The basic idea is that both Cardholders and Merchants must register with Certificate Authorities before they engage in transactions. Unsuitable

individuals (known criminals, for example) may not get past this stage. Reliable (or reliable-looking) principals can then engage in business. During the purchase phases, all parties commit themselves to each transaction by using digital signatures. In this way, registered Cardholders can make purchases without sharing account details with the Merchant.

3 The SET Purchase Protocols

The purchase phase is complicated, involving interaction among three parties and several alternative protocol paths. For instance, *Purchase Requests* may be signed or unsigned, depending upon whether the Cardholder has run the Registration phase. *Payment Authorization* may be invoked during Purchase Request, or authorizations may be batched for processing later. Other complications include split shipments, payment by instalments, frequent-flyer bonuses, car rental ratings and other frills.

Here, we simplify and combine Payment Authorization with Purchase Request, yielding in effect a six-step protocol. The version below is slightly simpler even than that modelled in Isabelle: certificates are omitted and the PKCS digital envelopes [17] are replaced by simple public-key encryption. Reducing the SET purchase phase to six messages has not been trivial. A number of tricky issues in the modelling are discussed elsewhere [3].

Initial Shopping Agreement. The Cardholder and Merchant agree upon the order description (OrderDesc) and the purchase amount (PurchAmt). This agreement step, called the *SET Initiation Process* in the *Programmer's Guide* [11, page 45], is not part of SET and occurs just before it.

Purchase Initialization Request. The Cardholder sends the Merchant a freshness challenge (Chall_C) and a local transaction identifier (LID_M).

$$1.\ C \to M : \text{LID\_M}, \text{Chall\_C}$$

Purchase Initialization Response. The Merchant replies with a signed message that includes a freshness challenge (Chall_M) and generates a nonce that serves as a globally unique transaction identifier[1] XID. Also returned is the public-key certificate of a Payment Gateway, which is determined by the Merchant's bank and the card brand.

$$2.\ M \to C : \mathsf{Sign}_{\mathsf{priSK}\,M}(\text{LID\_M}, \text{XID}, \text{Chall\_C}, \text{Chall\_M})$$

[1] "It is a randomly generated 20 byte variable that is globally unique (statistically)." [11, page 267].

Purchase Request. This is the most interesting message in SET. The Merchant and Payment Gateway must agree on the Cardholder's purchase, although each of them gets only partial information: the Merchant does not know the card details, and the Payment Gateway does not know what is being bought. To meet this objective, SET uses a *dual signature*. The Cardholder signs the concatenation of the hashes of the Payment Instructions and the Order Information. He combines this with the card details, including the PAN and other secret numbers, CardSecret and PANSecret, which help to authenticate him. Then he encrypts everything using the Payment Gateway's public key, $\mathsf{pubEK}\,P$. He sends this to the Merchant, along with the Order Information and the hash of the Payment Instructions. Much information is duplicated so that the various parties can confirm the hashes.

$$3.\ C \to M : \mathrm{PIDualSigned}, \mathrm{OIDualSigned}$$

Here, C has computed

$$\begin{aligned}
\mathrm{HOD} &= \mathsf{Hash}(\mathrm{OrderDesc}, \mathrm{PurchAmt}) \\
\mathrm{PIHead} &= \mathrm{LID_M}, \mathrm{XID}, \mathrm{HOD}, \mathrm{PurchAmt}, \mathrm{M}, \mathsf{Hash}(\mathrm{XID}, \mathrm{CardSecret}) \\
\mathrm{OIData} &= \mathrm{XID}, \mathrm{Chall_C}, \mathrm{HOD}, \mathrm{Chall_M} \\
\mathrm{PANData} &= \mathrm{PAN}, \mathrm{PANSecret} \\
\mathrm{PIData} &= \mathrm{PIHead}, \mathrm{PANData} \\
\mathrm{PIDualSigned} &= \mathsf{Sign}_{\mathsf{priSK}\,C}(\mathsf{Hash}(\mathrm{PIData}), \mathsf{Hash}(\mathrm{OIData})), \\
&\qquad \mathsf{Crypt}_{\mathsf{pubEK}\,P}(\mathrm{PIHead}, \mathsf{Hash}(\mathrm{OIData}), \mathrm{PANData}) \\
\mathrm{OIDualSigned} &= \mathrm{OIData}, \mathsf{Hash}(\mathrm{PIData})
\end{aligned}$$

An *unsigned* Purchase Request obviously lacks these interesting features and does not authenticate the Cardholder. Merchants may reject such requests.

Authorization Request. The Merchant seeks authorization from the Payment Gateway after receiving the Purchase Request. First, he verifies the dual signature, using the supplied hash of the Payment Instructions. He also verifies the Order Information. He takes the Payment Instructions (which he cannot read) and combines them with transaction identifiers and the hash of the Order Information. This he signs and encrypts using the Payment Gateway's public key.

$$4.\ M \to P : \mathsf{Crypt}_{\mathsf{pubEK}\,P}(\mathsf{Sign}_{\mathsf{priSK}\,M}(\mathrm{LID\_M}, \mathrm{XID}, \mathsf{Hash}(\mathrm{OIData}), \mathrm{HOD}, \mathrm{PIDualSigned}))$$

Authorization Response. The Payment Gateway verifies the dual signature using the supplied hash of the Order Information. He also compares certain hash values to check that the Cardholder and Merchant agree on the Order Description and Purchase Amount. The Payment Gateway can also verify the validity of the Cardholder's secret account information, using the Cardholder's certificate. If satisfied, he confirms authorization to the Merchant by signing a brief message containing the transaction identifier and purchase amount.

$$5.\ P \rightarrow M : \mathsf{Crypt}_{\mathsf{pubEK}\,M}(\mathsf{Sign}_{\mathsf{priSK}\,P}(\mathrm{LID\_M}, \mathrm{XID}, \mathrm{PurchAmt}, \mathrm{authCode}))$$

Purchase Response. The Merchant now sends a similar signed message to the Cardholder. It contains the hash of the Purchase Amount, which the Cardholder can verify. Disputes are resolved "out of band."

$$6.\ M \rightarrow C : \mathsf{Sign}_{\mathsf{priSK}\,M}(\mathrm{LID\_M}, \mathrm{XID}, \mathrm{Chall\_C}, \mathsf{Hash}(\mathrm{PurchAmt}))$$

In our model, we provide both signed and unsigned versions of Purchase Request, Authorization Request and Authorization Response.

A major difficulty how to model the initial shopping agreement. The *SET Initiation Process* is naturally "out of band": SET is concerned with payment, not with shopping. There are some suggestions in the SET *External Interface Guide* [8], but they cannot be considered part of the official protocol: the SET Initiation Process is not defined in the *Formal Protocol Definition*, and the *Programmer's Guide* [11, page 45] expects that "standards will be developed to address how this information is exchanged and how the SET protocol is initiated."

To prove that all parties agree on the details of a transaction at the end of a run, we must be precise about what transaction is being made at the start. Unfortunately, SET does not clearly specify how the Cardholder and Merchant identify their transaction. The *Programmer's Guide* states that the Merchant identifies the transaction from the identifier LID_M (if sent) or out of band otherwise [11, page 310]. After that, the parties use a different transaction identifier, XID: "XID is a transaction ID that is usually generated by the Merchant system, unless there is no [Purchase Initialization Response], in which case it is generated by the Cardholder system." [11, page 267].

We resolve this confusing state of affairs by (a) requiring the initial two messages to be present (so that the Merchant is responsible for generating XID appropriately), (b) using XID to identify transactions. We are experimenting with different ways of formalizing the initial bootstrapping phase and other researchers may make different choices.

Another point worth mentioning is the treatment of Authorization Response. In SET, the Payment Gateway always responds to the inquiries of the Merchant, even when authorization is denied. Thus, the authCode field may be a "yes", a "no", a "contact–human-at-800-SET-CARE" etc. For simplicity, our model assumes that only "yes" answers are returned (silent denial). It is worth asking whether the protocol is still secure when both "yes" and "no" answers are returned.

4 The Formal Model

We use the inductive method of protocol verification introduced by Paulson [14]. The operational semantics assumes an infinite population of honest agents obeying the protocol and a dishonest agent (the Spy) who can steal messages intended for other agents, decrypt them using any keys at his disposal and send new messages as he pleases. Some of the honest agents are compromised, meaning the Spy has full access to their secrets. A protocol is modelled by the set of all possible traces of events that it can generate. Events are of three forms:

- $\mathsf{Says}\, A\, B\, X$ means A sends message X to B.
- $\mathsf{Gets}\, A\, X$ means A receives message X.
- $\mathsf{Notes}\, A\, X$ means A stores X in its internal state.

Each agent has two asymmetric key pairs, one for signature and one for encryption. Apart from the Spy, agents are of four kinds:

- Certificate Authorities, which sign certificates for other agents, are written $\mathsf{CA}\, i$ (for $i \geq 0$).
- Cardholders are written $\mathsf{Cardholder}\, i$.
- Merchants are written $\mathsf{Merchant}\, i$.
- Payment Gateways, whose purpose is to pay Merchants, are written $\mathsf{PG}\, i$.

The *Root* Certificate Authority is $\mathsf{CA}(0)$ and the model assumes it to be uncompromised. Any other agents may be under the Spy's control. Protocol properties can usually be expected to hold only if the agents involved are uncompromised, but (as with most protocols) having compromised agents in the system should not compromise other transactions.

More details on our SET model appear elsewhere [3].

The formal specification of SET purchase messages in Isabelle is about 240 lines long, including some comments but excluding the general SET public-key model. Unsigned purchases add several rules to the specification: the unsigned purchase request itself, and also its handling by the Merchant and Payment Gateway.

Figure 1 presents part of this specification: the signed purchase request. Let us go through this description, not to explain every detail but to illustrate how a protocol step is modelled. The constant *set_pur* denotes a set of traces, and is the formal model of SET purchase.

Each protocol step consists of many preconditions (typically referring to previous messages being received or fresh keys being generated) and a postcondition (a new message sent). More precisely, the rule refers to a given trace, here called *evsPReqS*. The next two conditions, *C = Cardholder k* and *M = Merchant i*, define local abbreviations. The condition *CardSecret k* $\neq$ *0* checks that this Cardholder is registered, while *Key KC2* $\notin$ *used evsPReqS* and *KC2* $\in$ *symKeys* assert that *KC2* is a fresh symmetric key. The next several lines, starting with *HOD* and ending with *PIDualSigned*, express the dual signature. The *Gets C* event refers to the Cardholder's reception of the previous message. The following line refers to the transaction details agreed out of band with the Merchant. Skipping to the conclusion, we find the current trace being extended with a *Says* event for the dual signature.

Equations are seldom used in the inductive definition of academic protocols. They are necessary in SET because messages are long and many fields are repeated. Some repetition is inherent in the notion of dual signature, where one party receives fields in clear while another receives them hashed. Further repetition arises from the *EXcrypt* digital envelope, which the general SET model defines as follows:

```
EXcrypt K EK M m == {|Crypt K {|M, Hash m|}, Crypt EK {|Key K, m|}|}
```

Here *EK* is a public encryption key, *K* is a symmetric key, and *M* and *m* are fields. Note that *m* appears twice. Simplifying this to a simple public-key encryption would result in a considerable loss of precision[2] . Equations let us express this complex message succinctly. Unfortunately, Isabelle's simplifier expands them during proofs, producing subgoals many pages long. Handling such huge formulas requires additional memory and processor time,

[2]If we assume that private key are never compromised but symmetric keys can be lost or guessed, we are able to consider the potential case where the session key *K* is compromised and thus the confidentiality of *M* put at risk. By using public-key encryption under the same assumption this potential threat would be neglected by the formalization.

```
⟦evsPReqS ∈ set_pur;  C = Cardholder k;  M = Merchant i;
 CardSecret k ≠ 0;  Key KC2 ∉ used evsPReqS;  KC2 ∈ symKeys;
 HOD = Hash{|Number OrderDesc, Number PurchAmt|};
 PIHead = {|Number LID_C, Number XID, HOD, Number PurchAmt, Agent M,
            Hash{|Number XID, Nonce (CardSecret k)|}|};
 OIData = {|Number XID, Nonce Chall_C, HOD, Nonce Chall_M|};
 PANData = {|Pan (pan C), Nonce (PANSecret k)|};
 PIData = {|PIHead, PANData|};
 PIDualSigned = {|sign (priSK C) {|Hash PIData, Hash OIData|},
                  EXcrypt KC2 EKj {|PIHead, Hash OIData|} PANData|};
 Gets C (sign (priSK M)
              {|Number LID_C, Number XID,
                Nonce Chall_C, Nonce Chall_M,
                certCA (PG j) EKj 0 (priSK RCA)|})
   ∈ set evsPReqS;
 trans_details XID = {|Agent C, Agent M, Number OrderDesc,
                       Number PurchAmt|};
 Says C M {|Number LID_C, Nonce Chall_C|} ∈ set evsPReqS;
 Notes C (Number XID) ∉ set evsPReqS⟧
⟹ Says C M {|PIDualSigned, OIData, Hash PIData|}
    # Notes C (Number XID) # evsPReqS ∈ set_pur
```

Figure 1: Signed Purchase Request in Isabelle

and makes greater demands on the human verifier. SET's complexity is near the limit of what can be verified using our methods.

5 The Proofs

We followed the usual pattern suggested by Paulson [14]: possibility properties, regularity properties, secrecy properties. Finally, we proved guarantees for the SET participants, namely the Cardholder, Merchant and Payment Gateway. A guarantee for a agent can refer only to information available to that agent, such as messages it has sent or received.

Possibility properties affirm that the protocol can run from start to finish. And thus, for example, message formats are consistent between rounds. They are logically trivial, but due to the size of the rules can be non-trivial to verify. They say nothing about security, but constitute a vital sanity check on the protocol definition. For SET purchase, we proved possibility properties for both the signed and unsigned message flows.

Regularity properties are largely trivial. They include facts such as that private keys cannot become compromised during a run and that certificates signed by the Root Certification Authority are correct.

Under secrecy, first we must prove that the symmetric keys used in digital envelopes are secure. From this lemma, we can prove that nonces encrypted using those keys are secure.

More important is to prove guarantees for the various SET participants. The *Formal Protocol Definition* [10] does not formally specify the goals of SET. All we have are the explicit but imprecise requirements from the *Business Description* [9, page 6].

The first business requirement, confidentiality of the payment data, is satisfied: we have proved that both the PANSecret and the Cardholder's PAN remains secure. In one sense, these proofs require significant effort, since their proof scripts occupy a substantial part the entire proof script file. However, thanks to Isabelle's modularity and level of automation, we could easily adapt them from our previous work on the Registration phases [2].

For the remaining business requirements relevant to formal verification (2-4), we adopted as a general guideline that the Cardholder, Merchant and Payment Gateway should agree on all relevant details of the transaction. The Payment Gateway knows the Purchase Amount and credit card details. The Merchant knows about the order description and purchase amount. The Cardholder knows both sets of information.

Many of these guarantees involve verifying digital signatures and are easily proved by induction. Here are some examples.

- When the Merchant receives *Authorization Response*, he knows that the Payment Gateway signed it, including the transaction identifiers and the purchase amount, which the Merchant can separately confirm.

- When the Merchant sees the dual signature, he can check that it was intended for him and originated with the given Cardholder.

- When the Payment Gateway sees the dual signature, he can verify that the transaction is indeed between the given Cardholder and Merchant.

- When the Payment Gateway receives an Authorization Request with a dual signature, he knows that the Cardholder and Merchant agree on the essential details. The Purchase Amount is sent by only by the Cardholder, not by the Merchant. However, both parties separately compute the hash of the Order Description and Purchase Amount, and the Payment Gateway compares these hashes.

Equally important is what cannot be proved, which suggests potential vulnerabilities. We tried but failed to prove that the Cardholder and Payment Gateway agree on the latter's identity. Recall Lowe's [6] attack against the Needham-Schroeder public-key protocol: when Alice runs the protocol against Charlie, he can run the protocol with another agent, using Alice as an oracle to impersonate her. An analogous vulnerability exists in SET: the Cardholder does not sign anything that identifies which Payment Gateway is intended.

Could this vulnerability be exploited? One possibility is a denial-of-service attack. A rogue Payment Gateway could generate valid-looking Authorization Requests for another Payment Gateway by re-encrypting received requests using the latter's public key. The recipient would be forced to expend considerable resources, communicating with card issuers via financial networks, before it could detect that something was wrong. To be sure, we are speculating. But this vulnerability could easily be fixed by inserting the Payment Gateway's identity into PIData.

There is a related problem: the Cardholder relies entirely on the Merchant for the choice of the Payment Gateway. A bad Merchant, colluding with a bad Payment Gateway, could harvest account details by telling Cardholders always to use that particular Gateway. (We are well aware that SET allows designated merchants to receive some account details legitimately.)

Digital envelopes complicated the proofs. The simplified version of SET shown in §3 above just uses public-key encryption, but our Isabelle model is closer to SET itself: public-key encryption is applied to a symmetric key, which is used to encrypt the bulk of the message. We had to prove secrecy

of these symmetric keys, and the double encryptions caused case splits in subgoals. Also, we found it hard to prove that the symmetric keys were received intact. This may seem a peculiar thing to worry about, since these keys are part of the security mechanism and not part of the data being transmitted. Still, it would be odd if Alice sent a digital envelope involving key K and Bob received this envelope but involving key K'. These envelopes use hashing to establish a link between the two parts (see again page 7 for the definition of *EXcrypt*), but the key itself is not included in the hash; perhaps it should be.

To give you an impression of the notation, here is how the secrecy of the PAN is expressed in Isabelle:

```
⟦Pan (pan C) ∈ analz(knows Spy evs);  C = Cardholder k;
 CardSecret k ≠ 0;  evs ∈ set_pur⟧
⟹ ∃j M KC2 K ps X Y Z.
     Says C M {|{|X, EXcrypt KC2 (pubEK(PG j)) Y {|Pan(pan C),ps|}|}, Z|}

       ∈ set evs  ∧
     (PG j) ∈ bad
```

The main premise of this theorem is that a cardholder's PAN is available to the Spy. The conclusion is that, in the past, the cardholder must have sent a Purchase Request message involving a bad Payment Gateway. The proof is by induction and simplification. It relies on a lemma involving PANs that is itself proved by similar methods.

6 Related Work and Conclusions

Until now, the most complex protocols analyzed using the inductive method were Kerberos IV [4], TLS (the successor to SSL) [15], and the Cardholder Registration Phase of SET [2]. The verification of the Purchase Phase has still been an open problem.

People have used other methods. Meadows and Syverson [12] have proposed a language for describing SET specifications but have not actually verified the protocol. They have used the temporal language NPATRL (the NRL Protocol Analyzer Temporal Requirements Language) for specifying a number of SET's requirements. Some requirements are more technical, such as "honest principals will faithfully execute the protocol", others concern more closely the protocol goals. The paper is not about verifying those requirements, which is left as future work. Instead, it concentrates on the difficulties in specifying them formally, an issue that concerns us too.

Kessler and Neuman [5] have extended an existing belief logic with predicates and rules to reason about *accountability*. (Although accountability is not among the stated goals of SET, it is clearly desirable.) They concentrate upon the Merchant's ability to prove to a third party that the Order Information originated with the Cardholder. Using the calculus of the logic, they conclude by pen and paper that the goal is met, so the Cardholder cannot repudiate his having initiated the transaction. Equivalently, we have proved that the dual signature being in the traffic implies that the Cardholder sent it. Stoller [18] has proposed a theoretical framework for the bounded analysis of e-commerce protocols but has only considered an overly simplified description of the payment phase of SET. Lin and Lowe [7] have also independently proposed a general theory to take complex protocols and map them into simpler model checkable protocol. However, they limited their actual analysis to the Cybercash protocol.

We succeeded in analyzing an abstract, but still highly complex, version of the SET purchase protocols. Novel techniques were not required; the difficulty consisted in digesting the specification and scaling up. SET's dual signatures were found to work. The necessary repetition of fields generated huge expressions and rendered the proofs harder. We found no major protocol flaws, but a lack of explicitness make the proofs more difficult than they should have been, while weakening the eventual guarantees.

Acknowledgements. F. Massacci was supported by CNR and MURST grants. Part of this work was done while P. Tramontano was visiting the Computer Laboratory in Cambridge under a visiting scholarship for masters students by the University of Roma I "La Sapienza." The work at Cambridge was funded by the EPSRC grant GR/R01156/R01 *Verifying Electronic Commerce Protocols*.

References

[1] M. Abadi and R. M. Needham. Prudent engineering practice for cryptographic protocols. *IEEE Trans. on Software Engineering*, 22(1):6–15, January 1996.

[2] G. Bella, F. Massacci, L. C. Paulson, and P. Tramontano. Formal verification of cardholder registration in SET. In F. Cuppens, Y. Deswarte, D. Gollman, and M. Waidner, editors, *Computer Security — ESORICS 2000*, LNCS 1895, pages 159–174. Springer, 2000.

[3] G. Bella, F. Massacci, L. C. Paulson, and P. Tramontano. Issues in modelling the SET protocol, 2001. in preparation.

[4] G. Bella and L. C. Paulson. Kerberos version IV: Inductive analysis of the secrecy goals. In Quisquater et al. [16], pages 361–375.

[5] V. Kessler and H. Neumann. A sound logic for analysing electronic commerce protocols. In Quisquater et al. [16].

[6] G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using CSP and FDR. In T. Margaria and B. Steffen, editors, *Tools and Algorithms for the Construction and Analysis of Systems: second international workshop, TACAS '96*, LNCS 1055, pages 147–166. Springer, 1996.

[7] G. Lowe and M. Lin Hui. Fault-preserving simplifying transformations for security protocols. *J. of Comp. Sec.*, 9(3-46), 2001.

[8] Mastercard & VISA. *SET Secure Electronic Transaction: External Interface Guide*, May 1997. Available electronically at `http://www.setco.org/set_specifications.html`.

[9] Mastercard & VISA. *SET Secure Electronic Transaction Specification: Business Description*, May 1997. Available electronically at `http://www.setco.org/set_specifications.html`.

[10] Mastercard & VISA. *SET Secure Electronic Transaction Specification: Formal Protocol Definition*, May 1997. Available electronically at `http://www.setco.org/set_specifications.html`.

[11] Mastercard & VISA. *SET Secure Electronic Transaction Specification: Programmer's Guide*, May 1997. Available electronically at `http://www.setco.org/set_specifications.html`.

[12] C. Meadows and P. Syverson. A formal specification of requirements for payment transactions in the SET protocol. In R. Hirschfeld, editor, *Proceedings of Financial Cryptography 98*, volume 1465 of *Lecture Notes in Comp. Sci.* Springer-Verlag, 1998.

[13] A. Paller. Alert: Large criminal hacker attack on Windows NTE-banking and E-commerce sites. On the Internet at `http://www.sans.org/newlook/alerts/NTE-bank.htm`, Mar. 2001. SANS Institute.

[14] L. C. Paulson. The inductive approach to verifying cryptographic protocols. *J. of Comp. Sec.*, 6:85–128, 1998.

[15] L. C. Paulson. Inductive analysis of the internet protocol TLS. *ACM Trans. on Inform. and Sys. Sec.*, 2(3):332–351, 1999.

[16] J.-J. Quisquater, Y. Deswarte, C. Meadows, and D. Gollmann, editors. *Computer Security — ESORICS 98*, LNCS 1485. Springer, 1998.

[17] RSA Laboratories. *PKCS-7: Cryptographic Message Syntax Standard*, 1993. Available electronically at `http://www.rsasecurity.com/rsalabs/pkcs`.

[18] S. D. Stoller. A bound on attacks on payment protocols. In *Proc. 16th Annual IEEE Symposium on Logic in Computer Science (LICS)*, June 2001.