The Verification of an Industrial Payment Protocol

The SET Purchase Phase

Giampaolo Bella*
Computer Laboratory,
University of Cambridge
15 JJ Thomson Avenue,
Cambridge CB3 0FD (UK)
gb221@cam.ac.uk

Fabio Massacci
Dip. di Informatica e Telecom.
Università di Trento
via Sommarive 14,
38050 Povo (Trento) — Italy
massacci@ing.unitn.it

Lawrence C. Paulson
Computer Laboratory,
University of Cambridge
15 JJ Thomson Avenue,
Cambridge CB3 0FD (UK)
lcp@cl.cam.ac.uk

ABSTRACT

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

This paper reports the first verification results for the complete purchase phase of SET. Using Isabelle and the inductive method, we showed that the credit card details do remain confidential and customer, merchant and bank can confirm most details of a transaction even when some of those details are kept from them. The complex protocol construction makes proofs more difficult but still feasible.

Though enough goals can be proved to give confidence in SET, a lack of explicitness in the dual signature makes some agreement properties fail: it is impossible to prove that the customer meant to sent his credit card details to the payment gateway that receives them.

Categories and Subject Descriptors

C.2.0 [**Computer and Communication Networks**]: General—*Security and Protection*; C.2.2 [**Computer and Communication Networks**]: Network Protocols—*Protocol verification*; D.2.4 [**Software Engineering**]: Software/Program Verification—*Formal Methods, Theorem Proving*; F.3.1 [**Logics and Meaning of Programs**]: Specifying and Verifying and Reasoning about Programs—*Mechanical verification, Specification techniques*

General Terms

Design, Economics, Security, Verification

Keywords

Electronic Commerce, Security Protocols, Inductive Specifications, Formal Verification, Isabelle Proof Assistant

1. INTRODUCTION

The last years have seen a substantial progress in the formal verification of security protocols. Detailed analysis of cryptographic primitives, verification of Internet standards, and substantial progress in the automation of model-checking and theorem-proving procedures for security verification have boosted a field which outsiders believe populated by "Yet-Another-Weakness-of-Needham-Schroeder" papers.

Though protocols like the Internet Key Exchange protocol [16], the Cybercash protocol [11], the TLS/SSL protocol [21] all yielded to automatic or semi-automatic tools, full verification of SET (the Secure Electronic Transaction protocol by Visa and Mastercard) has remained out of reach.

The Challenges of SET

Why is SET such a challenge for formal verification? The first hurdle is the sheer size of the documentation [12, 13, 14, 15] which takes over 1000 pages. The second, more substantial obstacle is the protocol's complexity. Academic protocols are typically short, straight-line programs; they seldom go beyond two levels of encryption and generate few secrets. Internet protocols such as IKE and TLS use cryptography rather sparingly compared to SET. SET has many features that make its verification hard:

- *Multiple nested encryptions and duplicate message fields* require abbreviations. Most proof tools must expand abbreviations in order to reason about them, but for SET the result is huge expressions.
- *Ubiquitous generation of random numbers and keys* cause a state-space explosion in model checking. The standard technique of allowing only a handful of nonces and keys would not even allow a single execution to complete, let alone two or more parallel ones.
- *Many alternative protocol paths* make it hard to single out the few key roles used either by manual analysis

*Current Address: Dip. di Matematica e Informatica, Università di Catania, Viale A. Doria 6, I-95125 Catania (Italy), Email: giamp@dmi.unict.it

(as in the strand space model) or by model-checkers to restrict the search space.

SET's use of alternative protocol paths is not bad design but is driven by real requirements. For example, security-aware customers may have pre-registered with a financial institution and thus secured their credit cards against the merchant's eyes. Other customers may decide to trust the merchant and thus be content with a transaction secured against the outside world. From a merchant's perspective, all customers should able to conclude a purchase, whether they bothered to pre-register or not.

The complex structure of SET makes it a benchmark for security protocol design and verification, whether or not it will be a commercial success. For example, PKCS digital envelopes [23] will be used in future protocols, and understanding what formal guarantees they offer is vital. It is a litmus test for the industrial applicability of verification techniques: it checks whether they can scale up to the point when direct manual analysis or bug-finding automatic tools no longer work.

Our Contribution

In this paper, we focus on the main *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. If successful, this mechanism can be used in other multi-party protocols, as it does not require complicated group-cryptography.

In other papers [2, 3], we have described modelling issues of the general SET protocol and the analysis of its registration phases. We have followed the guidelines set out in those papers for a careful simplification of SET to make its analysis tractable (the full protocol has hundreds of fields) while retaining the most important mechanisms. Our simplified version is still one of the most complex protocols ever to be analysed formally.

The present paper describes the verification of SET's *Purchase* phases with the inductive method and the Isabelle theorem prover. We found that, on the whole, dual signatures work: credit card details do remain confidential and still all parties can be sure that they are dealing with the same transaction, even if they have only partial information.

Yet, the dual signature omits an important field, violating Abadi and Needham's [1] explicitness principle. So some guarantees are weaker than they should be — particularly for the Payment Gateway, who is supposed to authorize transactions. It is impossible prove that the customer intended to share his credit card detail with the Payment Gateway participating in the protocol. A bad Merchant and Payment Gateway can conspire to fool an honest Payment Gateway to authorize a transaction not meant for him. This scenario is not a realistic attack, but it is a counterexample to a fundamental guarantee: that the Cardholder and Payment Gateway should agree on the latter's identity. A simple change to the protocol can fix this problem.

From a verification perspective, our result shows that the inductive method (supported by a powerful prover like Isabelle) can scale up to protocol as complex as SET. However, we feel we have reached the limit of tractability for our approach. Better automation or user interfaces are needed for more complex protocols or more detailed models of SET.

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 successful and failed proofs (§5 and §6) and conclude with a discussion of related work (§8).

2. SET OVERVIEW

Most Internet merchants use the SSL protocol to prevent eavesdroppers from learning customers' account details, adopting the classical idea that bad persons are always outsiders. This arrangement has two major limitations:

- customers must trust merchants to keep these details secure, and merchants may be dishonest or, more often, incompetent [19].
- merchants must trust customers, who do not sign anything, and have little protection from stolen cards or from customers who repudiate their purchases.

Visa and Mastercard designed the SET protocol to solve these problems by keeping sensitive information confidential and by authenticating Cardholders and Merchants [13, page 6]. To achieve these goals, SET 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 credit card number (called the *primary account number* or PAN), and of a secret nonce (PANSecret), with the same role of the PIN for physical cards.
- with *Merchant Registration* a Merchant registers both a signature key and an encryption key.
- *Purchase Request* allows Cardholders to place orders with Merchants.
- *Payment Authorization* follows or is combined with *Purchase Request*. It allows a Merchant to verify the Cardholder's details with a Payment Gateway, which authorizes the transactions.
- *Payment Capture* is used by Merchants for the actual fund transfer.

The key idea is that Cardholders and Merchants register with Certificate Authorities before making purchases. Known fraudsters may be blocked at this stage. Registered principals can then engage in business. During the purchase phases, all parties commit themselves to each transaction by using digital signatures and hashes. Combining hashes and signature in suitable ways, Cardholders can make purchases without sharing account details with the Merchant, and order information with the Payment Gateway.

3. THE SET PURCHASE PROTOCOLS

Before going into details, let us point out some distinctive features of the protocol design.

The first idea is to use *Digital Envelopes*. Asymmetric encryption is too slow to be used for anything other than key distribution. Symmetric encryption is fast and must handle

the bulk of the data. So, if Alice wants to send a long message M to Bob, she generates a symmetric key K, encrypts K using Bob's public key and encrypts the message M using K. Some types of digital envelope bundle K with an additional short message m or with the hash of M. Other types combine the hash of m with M before the encryption using K, and so on. The SET Books [15] and the PKCS Standard [23] discuss the implementation and the cryptographic security of digital envelopes.

The second idea is the *Dual Signature*. This combination of hashes and digital signatures lets several parties agree on a transaction without giving each of them a complete view of the transaction. It is a simple alternative to group protocols.

Suppose that Alice wants to sign two documents O (for order) and P (for Payment) but wants to show to Bob only the O part of the transaction and to Charlie only the P part. Then she sends to Bob O, the hash of P and signs the concatenation of the hash of P with the hash of O. Clearly Bob can verify the signature because he has the hash of P and can generate the hash of O. Then she sends to Charlie P and the hash of O, together with the same signature which he can verify. However Charlie doesn't know what is in O and Bob doesn't know what is in P. Yet they can check that Alice signed the same thing.

Using a further level of encryption we can use Bob to forward the message to Charlie: Bob receives O, the hash of P, and the dual signature, plus P encrypted with Charlie's public key (so that Bob cannot read it). Then Bob checks the signature as before and forwards to Charlie the hash of O, the dual signature and the encrypted part P. Charlie can remove the encryption layer and verify the signature. Still Bob doesn't know P and Charlie doesn't know O.

Both digital envelopes and dual signatures complicate symbolic analysis by introducing excessive duplications. The symbolic hash of M is not shorter than M, though it may have different properties. Sending the symmetric encryption of M with the asymmetric encryption of the hash of M means that either the symbolic term describing the message M is duplicated. To make the presentation readable — both below and in the formal specifications — we introduce many abbreviations. But the messages blow up dramatically when abbreviations are unrolled.

The actual purchase phase is far more complicated than the description above. It involves interaction among three parties and several alternative protocol paths. *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 and payment by instalments.

Here, we combine Payment Authorization with Purchase Request, yielding in effect a six-step protocol. For sake of readability, the version below is simpler even than that actually modelled and verified in Isabelle: certificates are omitted and the PKCS digital envelopes [23] 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].

3.1 Initial Shopping Agreement

The Cardholder and Merchant agree on the order description (OrderDesc) and the purchase amount (PurchAmt). This agreement step, called the *SET Initiation Process* in the *Programmer's Guide* [15, page 45], is not part of SET and occurs just before it.

3.2 Purchase Initialization Request

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

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

3.3 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 (but omitted below) is the public-key certificate of a Payment Gateway, which is determined by the Merchant's bank and the card brand. In our formalization, a certificate is merely a message containing an agent's name and public key, signed by the Root Certification Authority [3].

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

3.4 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 PIData and the Order Information OIData. He combines this with the card details PANData, 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.

$$3.\ C \to M : \text{PIDualSign}, \text{OIDualSign}$$

Here, C has computed

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

An *unsigned* Purchase Request obviously lacks this combination of digital signatures and hashing. It authenticates the Cardholder using the hash of the PANSecret. Though it does not offer the guarantees of a digital signature, it is still better than sending the credit card details to the Merchant.

[1] "a randomly generated 20 byte variable that is globally unique (statistically)" [15, p.267].

3.5 Authorization Request

The Merchant seeks authorization from a Payment Gateway after receiving the Purchase Request. First, he verifies the dual signature, using the hash from 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.

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

3.6 Authorization Response

The Payment Gateway verifies the dual signature using the hash from the Order Information; checks that the Cardholder and Merchant agree on the Order Description and Purchase Amount by comparing certain hash values; and finally verifies 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.

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

3.7 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."

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

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

One issue is how to model the initial shopping agreement between the Cardholder and Merchant. 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. The *SET Initiation Process* is "out of band": SET is concerned with payment, not with shopping. There are suggestions in the SET *External Interface Guide* [12], but they are not part of the official protocol: the SET Initiation Process is not defined in the *Formal Protocol Definition*, and the *Programmer's Guide* [15, page 45] expects that "standards will be developed to address how this information is exchanged and how the SET protocol is initiated."

SET's system of transaction identifiers is elaborate. The *Programmer's Guide* states that the Merchant identifies the transaction from the identifier LID_M (if sent) or out of band otherwise [15, 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." [15, page 267]. In the latter case, the Merchant identifies the order by scanning the order description according out of band agreements.

We resolve this complicated 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 have tried various 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 actual authCode field may be a "yes", a "no", a "contact-human-at-800-SET-CARE" etc. For simplicity, our model assumes that principals only return "yes" answers and otherwise abandon the session. Other researchers might analyse the security of the protocol when both "yes" and "no" answers are returned.

4. THE FORMAL MODEL

We use the Isabelle theorem prover [18] with the inductive method of protocol verification introduced by Paulson [20]. 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.

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 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, though many compromised agents may be present. More details on our SET model appear elsewhere [3].

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 protocol step consists of many preconditions (typically referring to previous messages being received or fresh keys being generated) and a postcondition (a new message is sent or stored).

The purchase phase is specified in 240 lines of Isabelle text, including some comments but excluding the general SET public-key model (in total around 1000 lines). Unsigned purchases add several rules to the specification, namely the unsigned purchase request itself and its handling by the Merchant and Payment Gateway.

Figure 1 presents part of this specification: the signed purchase request. Let us go through it, not to explain every detail but to illustrate how a protocol step is modelled.

More precisely, the rule refers to a given trace, here called `evsPReqS`. The trace `evsPReqS` is a possible sequence of events

```
⟦evsPReqS ∈ set_pur;  C = Cardholder k;  CardSecret k ≠ 0;  Key KC2 ∉ used evsPReqS;  KC2 ∈ symKeys;
 Transaction = {|Agent M, Agent C, Number OrderDesc, Number PurchAmt|};
 HOD = Hash{|Number OrderDesc, Number PurchAmt|};
 OIData = {|Number LID_M, Number XID, Nonce Chall_C, HOD, Nonce Chall_M|};
 PIHead = {|Number LID_M, Number XID, HOD, Number PurchAmt, Agent M, Hash{|Number XID, Nonce (CardSecret k)|}|};
 PANData = {|Pan (pan C), Nonce (PANSecret k)|};
 PIData = {|PIHead, PANData|};
 PIDualSign = {|sign (priSK C) {|Hash PIData, Hash OIData|}, EXcrypt KC2 EKj {|PIHead, Hash OIData|} PANData|};
 OIDualSign = {|OIData, Hash PIData|};
 Gets C (sign (priSK M){|Number LID_M, Number XID,Nonce Chall_C, Nonce Chall_M,
                  cert P EKj onlyEnc (priSK RCA)|})∈ set evsPReqS;
 Says C M {|Number LID_M, Nonce Chall_C|} ∈ set evsPReqS;
 Notes C {|Number LID_M, Transaction|} ∈ set evsPReqS⟧
⟹
 Says C M {|PIDualSign, OIDualSign|} # evsPReqS ∈ set_pur
```

Figure 1: Signed Purchase Request in Isabelle Syntax

that happened so far. The constant *set_pur* denotes the set of traces belonging to the SET purchase. So by *evsPReqS* ∈ *set_pur;* we simply denote the fact that this trace must belong to the set of traces of the SET Purchase.

The next condition, *C = Cardholder k*, defines a local abbreviation: *C* stands for the k-th Cardholder. In the actual SET protocol he would be the principal responsible for taking the action described in this rule. Recall that we have no limit to the number of cardholders. The condition *CardSecret k* ≠ *0* checks that this k-th Cardholder is registered: the *CardSecret* field belongs to the certificates exchanged by the Merchant and the Cardholder. It is fixed to 0 if the Cardholder didn't bother to register his public key with a certification authority. The conditions *Key KC2* ∉ *used evsPReqS* and *KC2* ∈ *symKeys* say that *KC2* is a fresh symmetric key.

The next line, *Transaction = ...* refers to the transaction details agreed out of band by the Cardholder and the Merchant. The next several lines, starting with *HOD* and ending with *OIDualSign*, express how the dual signature is constructed by the Cardholder and are taken from the protocol description.

The relevant events that should have happened along this trace are now described: the Cardholder agreed out-of-band with the Merchant about the transaction details (the *Notes C* event), he sent the Purchase Initialization Request to the Merchant (the *Says C M* event) and finally the *Gets C* event refers to the Cardholder's reception of the Purchase Initialization Response. These steps are those described in the actual SET protocol (see Section 3).

Skipping to the conclusion, we find the current trace being extended with a *Says C M* event for the dual signature.

In all protocols verified in the past, equations have seldom been used with the inductive method. They are necessary in SET because messages are long, with many repeated fields — as we have previously noted. A message containing both *M* and *Hash M* involves a repetition of *M* when it is treated formally. 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 (as we have done in our informal presentation of the SET Purchase Phase on Section 3) would result in a considerable loss of precision. The digital envelope not only admits the possibility of the symmetric key's being compromised, but binds the message components more loosely.

Indeed, if we model the digital envelope by the direct encryption *Crypt EK {|M, m|}*, then both *M* and *m* can only be compromised together. In the actual model, one can lose the symmetric key, thus disclosing *M* to the spy, without disclosing *m*. This difference in the security level is also witnessed by the difference of importance between *M* and *m* in the actual SET protocol. The message *M* contains the order information (tough luck if the spy discovers your bad taste in music), while *m* contains the credit card details (deserving of stronger protection).

Our first experiments made this very simplification. They were useful in discovering tricky points and identifying the main guarantees to be proved. The current formalization includes full digital envelopes, which makes the proofs considerably more complicated.

During the modelling stage, the support for equational reasoning in Isabelle allows to express messages like Purchase Request succinctly. Unfortunately, Isabelle's simplifier expands equations during proofs, producing subgoals many pages long. Handling such huge formulas requires additional memory and processor time, and makes great demands on the human verifier. SET's complexity is near the limit of what can be verified using this method.

5. VERIFIED PROPERTIES

The *Formal Protocol Definition* [14] does not formally specify the goals of SET. All we have are the explicit but imprecise requirements from the *Business Description* [13, page 6] which we quote here:

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

We followed the usual pattern suggested by Paulson [20]: possibility properties, regularity properties, secrecy properties. Finally, we proved guarantees for the SET participants,

namely the Cardholder, Merchant and Payment Gateway. A guarantee for an 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 end. 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, essential for protocols as complex as SET: the protocol may be secure simply because it cannot be run. For the Purchase transaction, we proved possibility properties for both the signed and unsigned message flows.

Regularity properties are simple properties of the model, proved by induction: private keys cannot become compromised during a run, certificates signed by the Root Certification Authority are correct, etc

As for *secrecy properties*, 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. Using these two lemmas we can show that the first business requirement — confidentiality of the payment data — is satisfied. In particular we have proved that both the PANSecret and the Cardholder's PAN remain secure. In one sense, these proofs require significant effort, since their proof scripts occupy a substantial part the proof script. However, thanks to Isabelle's level of automation, we proved them easily by building on the work from the Registration phases [4]. Here is one example, stated informally and again using Isabelle syntax (in Figure 2).

THEOREM 1 (PAN SECRECY, SIGNED). *If the Spy can get the PAN of a registered Cardholder, then the Cardholder has previously issued a Purchase Request involving a bad Payment Gateway.*

Let us see this theorem as stated formally in Figure 2 to give a glimpse of the way properties are specified in isabelle. By `analz(knows Spy evs)` we denote the set of all messages that the Spy can deduce from his knowledge of the events of the trace evs. The inequality `CardSecret k ≠ 0` expresses that `Cardholder k` is registered. The existentially quantified variables refer to the unknown (and irrelevant) items that compose the Purchase Request sent to the bad Payment Gateway (called `P`). Check again the format of the Signed Purchase Request in section 3 and in Figure 1: here the only relevant detail is `PANData`.

The proof, a typical confidentiality argument, involves induction followed by heavy equational simplification and the automatic elimination of trivial cases. It relies on a lemma on PANs that is proved by similar methods. The version for unsigned cardholders must be stated and proved separately because the Purchase Request message has a different format.

For the remaining business requirements relevant to formal verification (2–4 in the list above), 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.

Most of these guarantees involve verifying digital signatures. Some of them also apply to unsigned purchases. On the whole, they are easily proved by induction, sometimes relying on lemmas also proved by straightforward inductions. The main problem is coping with the size of the formulas. Here are the main results. Note that assumptions of the form `A ∉ bad` state that agent `A` is uncompromised. (In our model, *compromised* means under the Spy's control.) In some cases they are reasonable: when stating a guarantee for a Merchant it is realistic to assume that the Merchant himself is uncompromised. In this case we do not mention it. In all other cases they mark an important assumption on our trust model: who, beside us, must be trusted for the protocol to be secure.

THEOREM 2. *When the Merchant receives* Authorization Response *from a trusted Payment Gateway, he knows that the Payment Gateway signed it, including the transaction identifiers and the purchase amount, which the Merchant can separately confirm.*

The corresponding text in Isabelle syntax is in Figure 3. Notice that in Figure 3 we have the condition `PG j ∉ bad`: if the Merchant can trust the j-th Payment Gateway to be uncompromised (in both his data and his private keys) then Authentication Responses allegedly from the j-th Payment Gateway do indeed come from him.

THEOREM 3. *When the Merchant sees a dual signature from an uncompromised Cardholder, he can check (using `XID`) that it was intended for him and was issued by the Cardholder.*

Notice that in Figure 4 we have the condition `C ∉ bad`: the Merchant must trust the Cardholder. This assumption may seem dubious, but it makes sense here. The Cardholder has completed the SET Registration phase, so he can claim the backing of a reputable financial institution. SET is not designed to protect merchants from criminals but to keep criminals out of the system.

The remaining main theorems are for the Payment Gateway and the Cardholder and the last steps of the protocol.

THEOREM 4. *When a Payment Gateway sees the dual signature, from uncompromised Cardholder and Merchant, he can verify that it originated with the given Cardholder for a transaction with the given Merchant. He can also verify that the Merchant intended him to handle the transaction.*

THEOREM 5. *When the Cardholder receives Purchase Response, from an uncompromised Merchant he knows that the Merchant sent it. He also knows that the Merchant received a signed message from a Payment Gateway chosen by the Merchant to authorize the purchase.*

6. FAILED PROPERTIES

Equally important is what cannot be proved, which suggests potential vulnerabilities. It is impossible to prove that the Cardholder and Payment Gateway agree on the latter's identity. Unless he trusts the Merchant, the Payment Gateway has no reason to believe that the Cardholder intended him to take part in the transaction. This lack of agreement occurs because the Cardholder does not sign anything that specifies the Payment Gateway.

If the original Payment Gateway is bad, and can collude with a bad Merchant, this scenario is possible: he can

```
⟦Pan (pan C) ∈ analz(knows Spy evs);  C = Cardholder k;  CardSecret k ≠ 0;   evs ∈ set_pur⟧
⟹
 ∃P M KC2 PIDualSign_1 PIDualSign_2 other OIDualSign.
  Says C M {|PIDualSign_1, EXcrypt KC2 (pubEK P) PIDualSign_2 {|Pan(pan C),other|}, OIDualSign|} ∈ set evs  ∧  P ∈ bad
```

Figure 2: PAN remains secret: Theorem 1 In Isabelle Syntax

```
⟦MsgAuthRes = {|Number LID_M, Number XID, Number PurchAmt|};
 Crypt (priSK (PG j)) (Hash MsgAuthRes) ∈ parts (knows Spy evs);   PG j ∉ bad;  evs ∈ set_pur⟧
⟹ ∃i KM HOIData HOD P_I.
 Gets (PG j) (EncB (priSK (Merchant i)) KM (pubEK (PG j)) {|Number LID_M, Number XID, HOIData, HOD|} P_I)
   ∈ set evs &
 Says (PG j) (Merchant i) (Crypt (pubEK (Merchant i)) (sign (priSK(PG j)) MsgAuthRes))
   ∈ set evs
```

Figure 3: M can trust AuthRes: Theorem 2 In Isabelle Syntax

remove the encryption from the dual signature; communicate the Cardholder's allegedly confidential data to the Merchant, who can then package everything back to an honest Payment Gateway. (In our model, "bad" is equivalent to having one's private keys compromised.)

We do not expect to see this attack in the real world, primarily because a bad Payment Gateway has more lucrative crimes to commit. No amount of tinkering with SET can reduce the need for absolute trust in Payment Gateways, who see the Cardholders' confidential account details. However, SET allows the Cardholder's software to abort the transaction *before* sending these confidential details if the proposed Payment Gateway is not certified by the same credit card company that issued the Cardholder's certificate [15, page 314]. This indicates that Cardholders are not expected to trust all Payment Gateways. It also confirms our view that that the name of the Payment Gateway is an essential element of the transaction. Since we require all parties to agree on all essential elements, we certainly want them to agree on the choice of Payment Gateway. The flaw can easily be fixed by inserting his identity into PIData.

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; recall the definition of `EXcrypt` near the end of §4. Again this problem is due to lack of explicitness: the key is not included in the hash, and it should be.

To summarize, the Payment Gateway can confirm neither the identity of the intended Payment Gateways nor the original symmetric key used in the Payment Information. This state of affairs is formalized as an ugly and fairly unsatisfactory theorem .

THEOREM 6. *When a Payment Gateway receives an Authorization Request with a dual signature, he knows that Cardholder and Merchant packaged a Payment Instruction (not necessarily the one just received) for some Payment Gateway (not necessarily him) with some digital envelope (not necessarily the one just opened) where they agreed on certain details that he can check. Purchase Amount is seen only by the Cardholder, not by the Merchant. However, both parties separately compute the hash of Order Description and Purchase Amount, and the Payment Gateway can compare them.*

7. IRRELEVANT PROPERTIES

Beside what failed and what succeeded, there are also other properties that are customarily proved for authentication protocols.

For instance one can scan Lowe's [10] or Gollmann's [6] classification and check what is verifiable. This is a tricky question because we eliminated fields that are immaterial to the main goals of the protocol but that may be essential for other security properties. For instance we have eliminated request-response identifiers which are recommended by Gong and Syverson [7] to make authentication protocols more robust and secure.

One may also argue that SET should also satisfy more advanced properties such as non repudiation. For instance, a Cardholder should be able to prove to a third party that a misbehaving CA tampered with the PANSecret. However, the verification of these properties implies major changes in the specifications and in the assumptions about the environment, and is likely to result in dubious proofs of security or highly debatable attacks.

8. RELATED WORK AND CONCLUSIONS

Until now, the most complex protocols analyzed using the inductive method were Kerberos IV [5], TLS (the successor to SSL) [21], 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 [17] 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 technical, such as "honest principals will faithfully execute the protocol," while others directly address the protocol

```
⟦MsgDualSign = {|HPIData, Hash OIData|};
 OIData = {|Number LID_M, etc|};
 Notes M {|Number LID_M, Agent P, extras|} ∈ set evs;
 Crypt (priSK C) (Hash MsgDualSign) ∈ parts (knows Spy evs);
 M = Merchant i;  C = Cardholder k;  C ∉ bad;  evs ∈ set_pur⟧
⟹ ∃PIData PICrypt.
     HPIData = Hash PIData &
     Says C M {|{|sign (priSK C) MsgDualSign, PICrypt|}, OIData, Hash PIData|} ∈ set evs
```

Figure 4: M can trust SignedPReq: Theorem 3 In Isabelle Syntax

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 Neumann [9] have extended an existing belief logic with predicates and rules to reason about *accountability*. Although accountability is not a stated goal 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 the transaction. Equivalently, we have proved that the dual signature being in the traffic implies that the Cardholder sent it. Stoller [24] 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. Hui and Lowe [11] have proposed a general theory to transform a complex protocol into a simpler protocol while preserving any faults. 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. The difficulty consisted in digesting the specification and scaling up. This is a major result: our methods scale to a level of complexity where intuition falters. However, the proofs often generated huge subgoals spanning several pages of text, stretching the human interaction with the prover.

Where do we go from here? The analysis of more complex protocols probably requires further advances, either in automation or in user interfaces. Isabelle is a general-purpose theorem prover; a specialized protocol verifier might be able to do better.

For example, the visualization of intermediate proof steps would be improved if we could avoid expanding abbreviations, with the attendant exponential blowup. No other protocols that we have seen make such heavy use of abbreviations, but they will become increasingly common as people try to verify industry standards out-of-the-box. Research is therefore needed on how to reason in the presence of abbreviations.

A theory of abstraction and compositionality might allow the proof of a big protocol to be divided into smaller parts. We should be able to separate the correctness proof for digital envelopes from that of a protocol that uses digital envelopes. Both automatic and manual verifiers would benefit. The problem of compositionality is being tackled by Guttman et al. [8, Section. 6] for manual analysis using strand spaces. The development and exploitation of such a theory is a major research problem, and we can expect any correctness proof for digital envelopes to impose conditions on how the protocol uses them. This still should lead to simpler proofs than at present, where we simply expand out the definitions of the envelopes.

The hardest task in our verification of SET has been that of digesting and abstracting the specifications. For us, the *Formal Protocol Definition* [14] is misleadingly named; it appears to consist of the *Programmer's Guide* [15] minus the information on how messages are handled. It should include explicit, formal statements of the protocol's goals. Complex protocols should be specified as refinements of more abstract protocols, whereas at present, protocol verifiers have to discover the abstract protocols themselves. It is a waste of effort, for the design must have evolved from an abstract protocol.

The myth that protocol verification is prohibitively expensive can be laid to rest. The cost of our efforts is small compared with the total cost of the SET design. The verification would have been cheaper and easier if we could have received the essential protocol directly from the designers. Verification should be an integral part of th design process.

From a security standpoint, it is customary to expect that every protocol is either correct or else vulnerable to attacks. However, SET lies in neither extreme. We were able to prove the most important goals, which gives grounds for reasonable confidence in SET. Yet, in the issue of agreement between the Cardholder and Payment Gateway on the latter's identity, we found that the property fails. This flaw is easy to fix.

Acknowledgements. F. Massacci was supported by CNR and MURST grants. The work at Cambridge was funded by the EPSRC grant GR/R01156/R01 *Verifying Electronic Commerce Protocols.*

9. 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, 2002. in preparation.

[4] G. Bella, F. Massacci, L. C. Paulson, and P. Tramontano. Verifying the SET registration protocols, 2002. submitted for publication.

[5] G. Bella and L. C. Paulson. Kerberos version IV: Inductive analysis of the secrecy goals. In Quisquater

et al. [22], pages 361–375.

[6] D. Gollmann. What do we mean by entity authentication? In *Proc. of the 15th IEEE Sym. on Sec. and Privacy*, pages 46–54. IEEE Comp. Society Press, 1996.

[7] L. Gong and P. Syverson. Fail-stop protocols: An approach to designing secure protocols. In *Proceedings of the 5th IFIP Working Conference on Dependable Computing for Critical Applications (DCCA-5)*, September 1995.

[8] J. Guttman. Security goals: Packet trajectories and strand spaces. In R. Focardi and F. Gorrieri, editors, *Foundations of Security Analysis and Design - Tutorial Lectures*, volume 2171 of *Lecture Notes in Comp. Sci.*, pages 197–261. Springer-Verlag, 2001.

[9] V. Kessler and H. Neumann. A sound logic for analysing electronic commerce protocols. In Quisquater et al. [22].

[10] G. Lowe. A hierarchy of authentication specifications. In *Proc. of the 10th IEEE Comp. Sec. Found. Workshop*, pages 31–43. IEEE Comp. Society Press, 1997.

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

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

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

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

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

[16] C. Meadows. Analysis of the Internet Key Exchange protocol using the NRL Protocol Analyzer. In *SSP-99*, pages 216–231. IEEE Comp. Society Press, 1999.

[17] 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.

[18] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Springer, 2002. LNCS Tutorial 2283.

[19] 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.

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

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

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

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

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