UEPS - A Second Generation Electronic Wallet

Ross J. Anderson

University of Cambridge Computer Laboratory,
Pembroke Street, Cambridge CB2 3QG, UK
rja14@cl.cam.ac.uk

UEPS, the Universal Electronic Payment System, is an electronic funds transfer product which is well suited to developing country environments, where poor telecommunications make offline operation necessary. It is designed around smartcard based electronic wallet and chequebook functions: money is loaded from the bank, via bank cards, to customer cards, to merchant cards, and finally back to the bank through a clearing system. This architecture is uniquely demanding from the point of view of security.

As far as we are aware, UEPS is the first live financial system whose authentication protocol was designed and verified using formal analysis techniques. This was achieved using an extension of the Burrows-Abadi-Needham [BAN] logic, and raises some interesting questions: firstly, such formal logics had been thought limited in scope to verifying mutual authentication or key sharing [GKSG]; secondly, our work has found hidden assumptions in BAN, and a problem with the postulates of the Gong-Needham-Yahalom logic [GNY], both concerning freshness; thirdly, we highlight the need for a formalism to deal with cryptographic chaining; and fourthly, this type of formal analysis turns out to be so useful that we believe it should be routine for financial and security critical systems.

1 Introduction

The OECD countries have many sophisticated networks which cater for autoteller machines, the use of credit and debit cards at the point of sale, interbank payments, and various other kinds of transaction. As the telecommunications infrastructure becomes ever faster and more reliable, these systems are increasingly online and centralised, and their existence can weaken the motive for introducing new crypto technologies such as smartcards.

Recent political developments have opened up the formerly centrally planned economies of Eastern Europe, India, Latin America and Africa to modern financial institutions and their associated payment systems. However, telecommunications are often a serious problem: decades of statism and neglect have left many of these countries with abysmal telephone networks, and villages are often without any connection at all. The lines that do exist are often not good enough to support fast modem communications or even the most basic autodial facilities. Transactions must often be carried out offline, and the risk of fraud with forged cards is such that the standard ISO magnetic stripe card with its associated PIN management techniques cannot be used.

This creates a one-off opportunity for these countries to leapfrog two generations in electronic payment technology, and go straight from manual ledger systems to distributed processing based on smartcard electronic wallets. Such systems can not only integrate retail banking and shopping functions but also provide the payment side of utility and government networks. The potential exists to slash transaction costs by more than an order of magnitude, and at the same time eliminate a major bottleneck in national economic development.

2 The Design Requirement

Our client, the Net 1 group, had secured a contract from the Permanent Building Society in South Africa to design and build a national eftpos system. This institution has the largest card base in the country with some 22% of the market, and most of its accounts were simple savings accounts. After building societies in SA were deregulated, plans were made to provide a full range of banking services.

However, poor telecommunications outside the major urban centres made it vital to carry out transactions offline. It was also necessary to be able to transact with utilities such as electricity distributors. Finally, as most customers have low incomes and had not previously used banking payment services, it was felt crucial to keep the transaction costs down, as charges similar to those made on traditional cheque accounts would have been significant compared with the customers' typical income and would have put them off using the system.

These requirements led inevitably to an electronic wallet approach, in which money is transferred between bank cards, customer cards and merchant cards using offline terminals, which can be made portable if necessary.

3 Previous Systems

First generation smartcard systems suffer from a number of drawbacks. One problem is that while the customer cards can be authenticated by a terminal, whether using challenge-response procedures or a public key algorithm, many vendors provide no such mechanism for the customer card to authenticate the terminal in turn. As a result, it may be possible to attack the card using a false terminal. One could, for example, try to record the card's data area at a time when it holds a large credit balance and then rewrite it later. Minimising the exposure to such frauds involves a number of host checks and other ad-hoc measures which are costly in processing terms and generally cumbersome and unsatisfactory.

Another problem is that banks prefer to use the familiar DES algorithm whose vulnerability to keysearch is well known [GO] and increasingly problematic. Several smartcards offer single DES encryption only, while we felt that double key encryption should be mandatory for payment systems.

The third problem is that existing regimes may result in a user carrying multiple cards, such as one for banking/eftpos, one for public telephones and one for the electricity meter. For both cost and strategic reasons we wanted a universal card which could be programmed to cater for multiple accounts on different systems, plus a secure means of transferring money between these accounts.

The final problem is integrity of design. Many existing financial systems have a long history of design evolution, and many frauds result from unforeseen loopholes which appear after seemingly unrelated changes. We felt it crucial to use formal methods to verify the correctness of the crypto protocol which forms the security kernel of the system.

4 Design Philosophy

The security of UEPS is based on two levels of authentication. The core is an electronic cheque which is generated by the client card, passed to the retailer card and then through a central clearing system to the customer's bank. The cheque carries two digital signatures: one generated with a key known only to the issuing bank's security module and the customer card, and one generated with a key which is controlled by the clearing house and loaded by them to the card before it is supplied to the bank. The latter signature is checked before funds are credited to the retailer presenting the cheque, while the former would only be checked in the event of a dispute. Both these signatures are standard message authentication codes, calculated on amount, payee and date.

Had public key technology been available, it would have been possible for a transaction recipient to check a signature. This was not an option at the time, and so we had to design a transaction protocol to prevent any bad cheques getting into the system in the first place. This uses challenge-reponse processes by which both cards in any transaction verify each other and carry on a secure session; it is similar to (but was developed independently of) the 'embedded observer' proposed by Chaum for electronic privacy applications [C]. In both cases, a trusted application in the smartcard vouchsafes for the authenticity of statements which the recipient cannot check directly.

The use of two independent security mechansims not only gives a high level of confidence, but also helps us meet audit and resilience requirements: to keep track of all the cash in the system on a day-by-day basis, and to guarantee that the compromise of any one key will not expose the whole system to fraud.

5 The Transaction Protocol

The transaction protocol is used to ensure the integrity of each step of the cash path, from bank to customer to merchant to clearer to bank. At each step, each transaction is made unique by random challenges, sequence numbers or both.

At the time UEPS was designed (1990-91), the only available programmable smartcard was the GemPlus product. This implements DES rather than a public key algorithm and we therefore had to base the cryptography on a transaction set between secure objects, a concept which is described in [DQ] and is already familiar in the banking industry.

Our first task, in view of our requirement for double encryption, was to implement a way of doing this which was within the technical constraints of the card. This was done by key chaining. Given a run of a protocol between two cards A and B, using a key pair, K1 and K2, and a series of message blocks A1, B1, A2, B2, ... we proceed as follows:

A $\longrightarrow$ B: K1(K2(A1))

B $\longrightarrow$ A: K1(K3(B1)) where K3 = K2(A1)

A $\longrightarrow$ B: K1(K4(A2)) where K4 = K3(B1)

In effect, the intermediate product of each double encryption is used as the second key in the following encryption. In this way each block doubles as an authenticator of all the previous messages in the protocol and information can be exchanged with maximum efficiency.

This is because one normally includes redundancy within each message, in the form of a fixed pattern or an already known variable, in order to check that the encryption has been performed with the right key. As we had a security parameter which dictated four bytes of redundancy, and a communications protocol which exchanged eight byte ciphertext blocks, a naive design would have resulted in half of each message being redundant. However, by key chaining we need only have one redundant data block, namely in the last message (which is the one which causes value to be credited in the recipient card).

During the crypto development process we had been using the Burrows-Abadi-Needham (BAN) logic to check the correctness of the authentication structure, and indeed it proved its value by highlighting several subtle errors and redundant fields in the first draft of our protocol specification. We found, however, that while the BAN logic supports conventional block chaining operations, it cannot deal with key chaining, although this seems to be just as good as data block chaining as a means of accumulating information in a cryptographic variable.

We will illustrate this by the exchange which takes place between a customer card C and a retailer card R when goods are purchased. The other transactions, whether bank - customer or retailer - bank, are essentially similar, but each transaction type uses different keys to ensure that no splicing attack can succeed with more than the probability of a random transaction.

Let C be the customer's name, N_C a nonce generated by him (a random number), R the retailer's name, N_R a nonce generated by him (the transaction sequence number), and X the electronic cheque. Then we can idealise the purchase transaction as:

1. $C \longrightarrow R : \{C, N_C\}_K \quad (= L)$

2. $R \longrightarrow C : \{R, N_R\}_L \quad (= M)$

3. $C \longrightarrow R : \{X\}_M$

In this protocol, the client card debits itself after step 2, while the retailer card only credits itself after step 4. The system is therefore failsafe from the bank's point of view: if anyone tampers with the protocol the only result they are likely to achieve is to increase the bank's float, by debiting the customer card without crediting any retailer.

In order to see how such a protocol can be validated, let us first consider a simplified protocol where the infomation is accumulated without chaining.

1*. $C \longrightarrow R : \{C, N_C\}_K$

2*. $R \longrightarrow C : \{R, N_R, C, N_C, \}_K$

3*. $C \longrightarrow R : \{C, N_C, R, N_R, X\}_K$

This can be analysed in a straightforward way using BAN. The trick is to start from the desired result and work backwards; in this case, we wish to prove that the retailer should trust the cheque, ie $R \mid\equiv X$. This will follow under the jurisdiction rule from $R \mid\equiv C \mid\Rightarrow X$ (R believes C has jurisdiction over X) and $R \mid\equiv C \mid\equiv X$ (R believes C believes X).

The former condition follows from the hardware constraint, that no-one except C could have uttered a text of the form $\{C, ...\}_K$. In effect, we have self-authenticating hardware.

The latter, that $R \mid\equiv C \mid\equiv X$, must be deduced using the nonce verification rule from $\sharp X$ (X is fresh) and $R \mid\equiv C \mid\sim X$ (R believes C uttered X).

Now $\sharp X$ follows from its occurrence in $\{C, N_C, R, N_R, X\}_K$ which contains the sequence number N_R, while $R \mid\equiv C \mid\sim X$ follows from the hardware constraint.

The BAN logic turns out to be easy to use because of the sparsity of its inference rules. When working back from a goal statement, it is rare for there to be more than one possible way to proceed. However, it provides no mechanism for dealing with the key chaining used in the actual protocol. In effect, we have to find a way of unravelling $\{X\}_{\{R,N_R\}_{\{C,N_C\}_K}}$ to $\{C, N_C, R, N_R, X\}_K$.

During the design of UEPS, we solved this problem by adding a further postulate to the BAN schema. The existing message meaning rule says that if P believes that the key K is shared with Q and sees a message X encrypted under K ($P \mid\equiv Q \leftrightarrow^K P, P \triangleleft \{X\}_K$), then he will believe that Q once said X ($P \mid\equiv Q \mid\sim X$).

To this we added a symmetrical rule to the effect that if P tries a key K to decrypt a block, and recognises the result as coming from Q ($P \mid\equiv Q \leftrightarrow^M P, P \triangleleft \{X\}_K$), then he will believe that Q in fact used K ($P \mid\equiv Q \mid\sim K$).

This however conflates recognition of messages with that of keys, and it has since been suggested that we might rather use the existing extension of the BAN logic by Gong, Needham and Yahalom [GNY] and Gong [G], which formalises the concept of recognisability. It turns out that axioms **F2** and **F7** in GNY (**F4** and **F14** in the later work by Gong) together imply a strange result: that someone who possesses a recognisable text X and a fresh key K may conclude that X is fresh by deducing first that $\{X\}_K$ is fresh, and then that X is.

The original BAN logic left the issue of freshness rules rather vague. On page eight it is stated that if (X, Y) is fresh, then X is; in other words, A can 'freshen' a statement X by concatenating it with a nonce just received from B, encrypting it with a key shared with B, and sending it to him.

Two questions follow: firstly, could we not analyse UEPS more easily by adding an extra freshness rule, which allows A to freshen X by encrypting it with a function of a nonce and a key? Secondly, is the word 'fresh' not misleading, in that we should rather concentrate on the action of uttering a statement?

GNY tries to make explicit many things that are left to 'common sense' in BAN, such as spotting redundancy in a decrypted message. It would seem that this refinement, if desirable, has not been thorough enough, and it may have to be extended further to distinguish the utterances of different parties. After all, the utility of 'freshness' is the knowledge it gives us of others' recent utterances.

However, there are good reasons to prefer a small set of rules. We found it more tedious to work with GNY than with BAN, as there are many more rules which have to be considered at each stage; logics which reason about belief and implication in the same calculus may fall foul of the transitivity paradoxes [H]; and finally, as noted above, a complex logic can have unforeseen consequences of a more mundane kind.

Another extension of BAN has been proposed by Kailar and Gligor [KG], who look at a sequence of messages M_i in the context of analysing a multiparty protocol.

This points to another possible approach: linked lists. It is likely that many protocols will prevent splicing attacks by some kind of chaining, so that message M_i contains a hash of message M_{i-1}. Authentication logics should be able to cope with this; a rule that in the linked list $\{A; B; C; D\}$ the freshness of utterance B would imply that of D (but not of A) would seem be right, and this should be a rule for both block and key chaining.

Our work suggests that there are two practical problems for future research in this field. Firstly, what granularity is desirable in our formal calculus; how much should be proved, and what should be left to common sense? Secondly, for a given granularity, what is the minimal effective set of postulates?

To return now to UEPS, we find that the validation, however it is performed, shows that the customer does not receive any confirmation of the transaction, but merely the knowledge that a valid retailer card is present. The value of this knowledge is to rule out a denial-of-service attack by a false terminal; but if the client bank is prepared to tolerate such attacks, then the first message of the protocol could be omitted.

One could also add a confirmation message from the retailer's card, but this would not solve the problem of an interruption after step 2, at which time the customer card has already committed to the transaction and debited itself, while the retailer has still not got the cheque.

As we have seen, no financial benefit can be gained by doing this, and accidents are sufficiently unlikely that they can be dealt with manually. The procedure is to refund to the customer any missing amount which remains unbanked after 21 days; but if the money were to be banked, the dispute would be resolved by comparing the two card records, or inspecting the paper tally roll printed by the merchant terminal (which shows the transaction plus a MAC). No dispute has been reported to date.

6 Practical Aspects

The average float of about ten days has turned out to be sufficient to cover the capital costs, so UEPS has funded itself out of cash flow. It offers the bank the same level of information and control as on a cheque account, as the clearing system tracks the daily balance of every participant. It gives value when the cheque is presented by the merchant, which is often one day after the purchase, and thus about two days faster than with a paper cheque; this does not lead to public complaints as the charges are an order of magnitude less than with the paper cheque system.

No losses were recorded during the first year of operation. The client institution considers it to be a success and the rate of both customer and merchant enrolment is being rapidly increased. From the customers' point of view, it makes fast, secure and low cost payment services available everywhere.

7 Conclusions

The BAN family of logics are not restricted to verifying mutual authentication and key exchange, but are a practical and helpful tool in the design of working crypto protocols, even (and especially) for substantial and complex systems. Although more elaborate systems (like GNY) exist, a first validation should be carried out with BAN, as it is easy to do, and failure to establish a desired result will indicate either a bug in the protocol, or something which BAN cannot express. It will normally be obvious what to do next.

We believe that formal verification should be routine in the design of financial and other security critical systems. It may well not be practical to verify all of a large banking software suite, and some aspects of system security (such as the disaster recovery plan) may remain forever beyond the reach of a formal calculus, but great benefit can be achieved for a small amount of effort by performing a formal analysis of the kernel of the system.

Even within the context of that kernel, we found the BAN logic to be useful as a design discipline. It did much more than help us tighten up the protocol security: it pointed out redundant fields in the messages, allowing the protocol to be made more efficient; it made clear the security dependencies; it provided intellectual stimulation at meetings with designers and programmers who were forced to examine and defend their assumptions; and finally, it greatly strengthened the client's confidence in the system.

8 Acknowledgements

The author is grateful to Roger Needham of the Computer Laboratory, to Serge Belamant of Net1 International, Argèles sur Mer, France, and to André Mansvelt and Gavin Shenker of Net1 Products, Johannesburg, for many stimulating discussions concerning this project.

References

[BAN] M. Burrows, M. Abadi and R. Needham, "A logic of Authentication", Report **39**, Digital Systems Research Center, Palo Alto, Ca.

[C] D. Chaum, "Achieving Electronic Privacy", in *Scientific American*, **267** no 2, August 1992, pp 76 - 81

[DQ] Y. Desmedt and J.-J. Quisquater, "Public-key Systems Based on the Difficulty of Tampering', in *Advances in Cryptology - CRYPTO 86*, Springer Lecture Notes in Computer Science **263** pp 111 - 117

[G] L. Gong, *Cryptographic Protocols for Distributed Systems* (PhD Thesis), University of Cambridge 1990.

[GKSG] V. D. Gligor, R. Kailar, S. Stubblebine and L. Gong, "Logics for Cryptographic Protocols - Virtues and Limitations", in *Proceedings, Computer Security Foundations Workshop* IV, IEEE 1991, pp 219 - 226

[GNY] L. Gong, R. M. Needham and R. Yahalom, "Reasoning about Belief in Cryptographic Protocols", in *Proceedings of the 1990 IEEE Computer Security Symposium on Research in Security and Privacy*, pp 234 - 248

[GO] G. Garon and R. Outerbridge, "DES Watch: An Examination of the Sufficiency of the Data Encryption Standard for Financial Institution Information Security in the 1990's', in *Cryptologia* XV no 3, July 1991, pp 177-193

[H] M. Hesse, *Structure of Scientific Inference*, Macmillan 1974, pp 142 - 146

[KG] R. Kailar and V. D. Gligor, "On Belief Evolution in Authentication Protocols", in *Proceedings, Computer Security Foundations Workshop* **IV**, IEEE 1991, pp 103 - 116