Proceedings of the 8th International Conference on Applied Informatics
Eger, Hungary, January 27–30, 2010. Vol. 1. pp. 295–302.

Applying Spi-calculus for PayWord*

László Aszalós[a], Andrea Huszti[b]

[a]University of Debrecen
laszalos@unideb.hu

[b]Hungarian Academy of Sciences and University of Debrecen
huszti.andrea@inf.unideb.hu

Abstract

Achieving strong security properties is a core part of wide acceptance of electronic commerce, hence it is essential to provide detailed security analysis for cryptographic protocols. There are protocols considered secure for a long time, still it is shown they contain flaws. Therefore it has been recognized that informal arguments about protocol correctness are not reliable. Formal methods offer a promising way for automated security analysis. Spi-calculus is designed for describing and analyzing cryptographic protocols. In this article we present how to employ this method for formalizing the micropayment scheme PayWord developed by Ronald L. Rivest and Adi Shamir.

Keywords: spi-calculus, micropayment, PayWord

MSC: 94A60, 68Q65

1. Introduction

Electronic commerce, especially electronic payment systems, deal with personal, confidential data, that should be protected against malicious attackers. Hence design of cryptographic protocols require special care, cryptographic primitives are applied to accomplish high level of security, to exclude adversaries' unauthorized access. Cryptographic schemes employing strong cryptographic tools, but weakly designed allow frauds. Thus substantial evaluation should be provided in order to prevent flaws. There are many methods to prove the safety properties of protocols.

One of the first attempts to analyze security protocols was the BAN logic [5]. Authors showed a flaw in the well known Needham-Schroeder key exchange proto-

*Research supported by the TARIPAR3 project grant Nr. TECH 08-A2/2-2008-0086, by OTKA grant K75566, Foundation of Action Austria Hungary No. 755u1. and by GOP-1.1.2-07/1-2008-0001 project. The work is supported by the TÁMOP 4.2.1./B-09/1/KONV-2010-0007 project. The project is implemented through the New Hungary Development Plan, co-financed by the European Social Fund and the European Regional Development Fund.

col, that was considered secure for a long time. Many researchers use general purpose model checkers [9, 12], others use theorem provers [3, 10] and strand spaces [14] for verifying security properties of cryptographic protocols. Recently other methods are popular: type checking [2], automatic protocol verifier [4, 8].

Our aim is to formalize a popular micropayment scheme called PayWord in a very strict manner, that can be a starting point of the substantial evaluation. As a formalization tool spi-calculus is applied, that provides all the cryptographic primitives PayWord requires. Our work is organized as follows. Section 2 describes elements of pi and spi-calculus, section 3 details the micropayment scheme PayWord, in section 4 we outline the steps of the scheme and give the precise programs of participants and finally we conclude our results.

2. Spi-calculus

The pi-calculus is a process calculus with first class channels to describe concurrent computations whose configuration may change during the computation [11]. Channels can be created and passed from one participant to the other. The pi-calculus does not provide cryptographic primitives, secure channels can be employed to transmit confidential data.

We use the following notations for terms: n: name, (L, M): pair, 0: zero, $succ(M)$: successor, x: variable, where L and M are terms. The channels may be restricted, so that only certain processes can communicate on them. We can define the following processes:

- $P|Q$ **concurrency**, where P and Q are processes
- $c(x).P$ **input prefixing**, process waiting for message sent on channel c before proceeding P, binding the message to name x
- $\overline{c}\langle x\rangle.P$ **output prefixing**, name x is emitted on channel c before proceeding P
- $!P$ **replication**, this process always can create a new copy of P
- $(\nu x)P$ **creation of a new name**, the process allocating a new constant x within P
- $\text{let}(x, y) = M$ in P **pair splitting**, behaves as $P[N/x][L/y]$ if term M is the pair (N, L), otherwise the process is stuck.
- 0 **nil process**, a completed and stopped process.

The spi-calculus is the extension of the pi-calculus [1]. It introduces primitives for cryptography, *i.e.* symmetric, asymmetric encryptions, digital signatures and hash functions:

- $\{M\}_N$ **shared key encryption**, encrypting M under the key N

- case L of $\{x\}_N$ in P **shared key decryption**, attempts decrypt the term L with the key N. If L is $\{M\}_N$ then the process behaves as $P[M/x]$, otherwise the process is stuck
- $\{[M]\}_{K^+}$ **public key encryption**, encrypting M under the key K^+
- case L of $\{[x]\}_{K^-}$ in P **public key decryption**, attempts decrypt the term L with the key K^-. If L is $\{[M]\}_{K^-}$ then the process behaves as $P[M/x]$, otherwise the process is stuck
- $[\{M\}]_{K^-}$ **signature generation**, encrypting M under the key K^-
- case L of $[\{x\}]_{K^+}$ in P **signature verification**, attempts decrypt the term L with the key K^+. If L is $[\{M]\}_{K^+}$ then the process behaves as $P[M/x]$, otherwise the process is stuck
- $[x$ is $M]P$ **testing**, if the value of x is M the process behaves as P, otherwise the process is stuck
- $H(x)$ **hash**, represents hash value of x

This enable us not just manipulate secure channels abstractly, but we can construct a secure channel by encryption of an insecure channel.

The spi-calculus use equations to describe the safety properties. Two processes are equivalent ($\simeq$) if no environment can distinguish them. Let the P' is the ideal version of P, *i.e.* all participants receive the original messages intended for them. We say the protocol satisfies the *authenticity* if $P \simeq P'$; and the protocol satisfies the *secrecy* if $P(M) \simeq P(M')$ for any message M' [1].

3. PayWord

By this time many kinds of Internet services and content providers have spread widely, that induce necessity of several payment systems. The production cost of content and services are often low and does not depend on the number of customers, hence these systems often charge very small amounts. Payment by credit cards require a minimal fee per transaction, therefore the service increase in cost significantly. Services usually employ micropayments are content providers that charge customers for on-line access to newspaper articles or . This motivates the introduction of Micropayment schemes, such as PayWord [13].

Design of micropayment schemes require special care concerning efficiency. The goal is to minimize the number of public key operations per payment, better to use hash functions instead.

In the scheme PayWord there are three participants: user (U), vendor (V) and broker (B). A user asks for the broker's authorization to make micropayments to the vendor, meaning we have a user-broker and a user-vendor relationship. Vendors initiate pay-off procedure with the broker, so we also have a broker-vendor

relationship. Among the three relationships the user-vendor one is short-term, the other two are long-term. User-vendor relationship occur ad-hoc and should be kept alive for short time, usually just few hours, and the on-line payment and delivery last only for seconds, but could be repeated several times. On the other hand bank-vendor and user-bank relationships are often off-line and these kind of transactions happen rarely, maximum once a day, hence computational cost is not so important.

Let us describe PayWord micropayment scheme in details. We employ digital signatures, where public and secret keys of the users, vendors and brokers are denoted by K_U^+, K_V^+, K_B^+ and K_U^-, K_V^-, K_B^-, respectively. We denote by $\{M\}_{K_i^-}$ the application of secret key on message M, where $i \in \{U, V, B\}$.

User-Broker relationship

The user U initiates a relationship with the broker B by requesting an authorized PayWord Certificate. U transports his credit-card number, the requested amount and public key K_U^+ to B on an authenticated encrypted channel.

$$1.\ U \to B : U, K_U^+$$

B generates U's certificate by signing digitally B, U, K_U^+ and E with key K_B^-. It means that broker B issues this certificate to user U, whose public key is K_U^+ and the certificate's expiration date is E. This certificate ensures ensures any vendor that valid amounts will be paid-off before date E.

$$2.\ B \to U : \{B, U, K_U^+, E\}_{K_B^-}$$

User-Vendor relationship

This relationship is short-term, meaning user buys an article with 5 pages and visits another website. We assume that the same user might come back later and requests another article on the same website. In case of first shopping U generates a *payword chain* $w_1, w_2, \ldots w_n$ in the following way. First U generates a random number[6, 7] w_n, then calculates

$$w_i = H(w_{i+1}),$$

where $i \in \{n-1, n-2, \ldots, 0\}$. We call $w_1, \ldots w_n$ paywords, w_0 is the root of the chain. U chooses the number n arbitrarily beyond the requested amount and generates a certificate containing the vendor's identification information V, his PayWord Certificate, w_0 as a commitment and the actual date D. This certificate is signed by the user's secret key K_U^-. The user should keep track of commitments he sent.

$$3.\ U \to V : \{V, \{B, U, K_U^+, E\}_{K_B^-}, w_0, D\}_{K_U^-}$$

The vendor verifies U's signature by public key K_U^+, and B's signature by K_B^+, checks whether D is before E and stores w_0 with the user information.

After sending the certificate U makes his payment. A payment is a pair of a payword and the corresponding index (w_i, i), where $i \in \{1, 2, \ldots, n\}$. It is important that the user sends his paywords starting from w_1, the w_2 and so on.

$$4.1\ U \to V : w_1, 1$$
$$4.2\ U \to V : w_2, 2$$
$$\vdots$$
$$4.n\ U \to V : w_n, n$$

Vendor V verifies the received payword w_i by applying i times the hash function on it, *i*.e. checks $H^i(w_i) = w_0$ in case of the first shopping. If U requests articles from the V not for the first time, then V will verify w_i with the stored payword w_j, where $j < i$, *i*.e. checks $H^{i-j}(w_i) = w_j$. By storing the payword with the highest index, V prevents double spending.

Vendor-Broker relationship

Vendor V sends all necessary information to B for pay-off. V transmits the certificate generated by U, the last payword received from U and the corresponding index.

$$5.\ V \to B : \{V, \{B, U, K_U^+, E\}_{K_B^-}, w_0, D\}_{K_U^-}, w_l, l$$

B verifies the signature of user's certificate with K_U^+, checks whether identity information received matches with V, the expiring date and validity of the payword, *i. e.* $H^l(w_l) = w_0$. If all verifications hold, then B pays the proper amount to V.

4. Formalization

In this section first we formalize all messages in protocol PayWord, in the following way: identifiers of steps (e.g. 1, $1'$) correspond to sequential numbers of the messages above, each step is an elementary action, *i.e.* sending, receiving a message, verifying a signature and testing values of variables. Table 1 lists all steps.

We are formalizing the steps on the Table 1 in Spi-calculus. We differentiate three programs: user's, bank's and vendor's program, describing all duties that the participants should do during the whole protocol, respectively.

User's program

Step	**Program**	**Remark**
1.	$\overline{C_{UB}}\langle U, K_U^+\rangle$	User sends its data
$2'$.	$C_{UB}(u)$	receives the Bank's answer
$2''$.	case u of $\{\!\{v\}\!\}_{K_B^+}$	checks the signature
	let $(v_1, v_2, v_3, v_4) = v$ in	
$2'''$.	$[v_1$ is $B][v_2$ is $U][v_3$ is $K_U^+]$	tests the data
3.	$\overline{C_{UV}}\langle\{\!\{V, u, H^n(M), D\}\!\}_{K_U^-}\rangle$	User sends the certificate
4.1.	$\overline{C_{UV}}\langle H^{n-1}(M), 1\rangle$	sends the first payword
	$\vdots$	
4.n.	$\overline{C_{UV}}\langle M, n\rangle$	sends the last payword

1.	U	$\rightarrow$		: U, K_U^+
1′.		$\rightarrow$	B	: x_1, x_2
2.	B	$\rightarrow$		: $[\{B, x_1, x_2, E\}]_{K_B^-}$
2′.		$\rightarrow$	U	: u
2″.			U	: signature check u as $[\{(v_1, v_2, v_3, v_4)\}]_{K_B^+}$
2‴.			U	: test $v_1 = B$, $v_2 = U$,$v_3 = K_U^+$
3.	U	$\rightarrow$		: $[\{V, u, w_0, D\}]_{K_U^-}$
3′.		$\rightarrow$	V	: p
3″.			V	: signature check p as $[\{(q_1, q_2, q_3, q_4)\}]_{K_U^+}$
3‴.			V	: signature check q_2 as $[\{(r_1, r_2, r_3, r_4)\}]_{K_B^+}$
3⁗.			V	: test $q_1 = V$, $r_1 = B$, $r_2 = U$, $r_3 = K_U^+$, $r_4 > q_4$
4.1.	U	$\rightarrow$		: w_1, 1
4.1′.		$\rightarrow$	V	: s_1, i_1
4.1″.			V	: test $q_3 = H(s_1)$, $i_1 = 1$
4.2.	U	$\rightarrow$		: w_2, 2
4.2′.		$\rightarrow$	V	: s_2, i_2
4.2″.			V	: test $s_1 = H(s_2)$, $i_2 = i_1 + 1$
			$\vdots$	
4.n.	U	$\rightarrow$		: w_n, n
4.n′.		$\rightarrow$	V	: s_n, i_n
4.n″.			V	: test $s_{n-1} = H(s_n)$, $i_n = i_{n-1} + 1$
5.	V	$\rightarrow$		: p, s_n, i_n
5′.		$\rightarrow$	B	: y_1, y_2, y_3
5″.			B	: signature check y_1, as $[\{z_1, z_2, z_3, z_4\}]_{K_U^+}$
5‴.			B	: signature check z_2, as $[\{t_1, t_2, t_3, t_4\}]_{K_B^+}$
5⁗.			B	: test $t_1 = t_1$, $x_1 = t_2$, $x_2 = t_3$, $z_3 = H^{y_3}(y_2)$

Table 1: Steps of PayWord

Bank's program

Step	**Program**	**Remark**
1′	$C_{UB}(x)$	Bank receives User's data
	let $(x_1, x_2) = x$ in	
2	$\overline{C_{UB}}\langle[\{B, x_1, x_2, E\}]_{K_B^-}\rangle$	sends his certificate back
5′	$C_{VB}(y)$	receives Vendor's request
	let $(y_1, y_2, y_3, y_4) = y$ in	
5″	case y_1 of $[\{z\}]_{K_U^+}$	checks User's signature
	let $(z_1, z_2, z_3, z_4) = z$ in	
5‴	case z_2 of $[\{t\}]_{K_B^+}$	checks Bank's signature
	let $(t_1, t_2, t_3, t_4) = t$ in	
5⁗	$[t_1$ is $B][x_1$ is $t_2][x_2$ is $t_3]$	tests the data
	$[H^{y_3}(y_2)$ is $z3]$	

Vendor's program

Step	Program	Remark
$3'$.	$C_{UV}(p)$	Vendor receives User's certificate
$3''$.	case p of $\{\!\{q\}\!\}_{K_U^+}$ in	checks User's signature
	let $(q_1, q_2, q_3, q_4) = q$ in	
$3'''$.	case q_2 of $\{\!\{r\}\!\}_{K_B^+}$	checks Bank's signature
	let $(r_1, r_2, r_3, r_4) = r$ in	
$3''''$.	$[q_1$ is $V][r_1$ is $B][r_2$ is $U]$	tests data
	$[r_3$ is $K_U^+][r_4 > q_4]$	
$4.1'$.	$C_{UV}(s_1, i_1)$	receives the first payword
$4.1''$.	$[q_3$ is $H(s_1)][i_1$ is $1]$	tests data
	$\vdots$	
$4.n'$.	$C_{UV}(s_n, i_n)$	receives the last payword
$4.n''$.	$[s_{n-1}$ is $H(s_n)][i_n$ is $i_{n-1} + 1]$	tests data
5	$\overline{C_{VB}}\langle p, s_n, i_n\rangle$	sends the bill to the Bank

5. Conclusion and further work

We have outlined the micropayment scheme PayWord, and a tool for describing cryptographic protocols called Spi-calculus. We have presented the scheme as a sequence of messages, and constructed the corresponding steps of elementary actions. Finally we have formalized the scheme in Spi-calculus. This is a milestone towards verifying cryptographic properties of the scheme applying an automated protocol checker.

References

[1] Abadi, M., and Gordon, A. D. A calculus for cryptographic protocols: The spi calculus. *Inf. Comput. 148*, 1 (1999), 1–70.

[2] Backes, M., Hritcu, C., and Maffei, M. Type-checking zero-knowledge. In *ACM Conference on Computer and Communications Security* (2008), P. Ning, P. F. Syverson, and S. Jha, Eds., ACM, pp. 357–370.

[3] Bella, G., Massacci, F., and Paulson, L. C. Verifying the set purchase protocols. *J. Autom. Reasoning 36*, 1-2 (2006), 5–37.

[4] Blanchet, B. Automatic verification of correspondences for security protocols. *Journal of Computer Security 17*, 4 (July 2009), 363–434.

[5] Burrows, M., Abadi, M., and Needham, R. M. A logic of authentication. *ACM Trans. Comput. Syst. 8*, 1 (1990), 18–36.

[6] Folláth, J. Construction of pseudorandom binary sequences using additive characters over GF(2^k). *Period. Math. Hungar. 57*, 1 (2008), 73–81.

[7] Herendi, T. Uniform distribution of linear recurring sequences modulo prime powers. *Finite Fields Appl. 10*, 1 (2004), 1–23.

[8] Kusters, R., and Truderung, T. Using proverif to analyze protocols with diffie-hellman exponentiation. *Computer Security Foundations Symposium, IEEE 0* (2009), 157–171.

[9] Lowe, G. Casper: A compiler for the analysis of security protocols. In *CSFW* (1997), IEEE Computer Society, pp. 18–30.

[10] Meadows, C. Language generation and verification in the nrl protocol analyzer. In *CSFW* (1996), IEEE Computer Society, pp. 48–61.

[11] Milner, R., Parrow, J., and Walker, D. A calculus of mobile processes 1. *Inf. Comput. 100*, 1 (1992), 1–40.

[12] Mitchell, J. C., Mitchell, M., and Stern, U. Automated analysis of cryptographic protocols using mur-phi. In *IEEE Symposium on Security and Privacy* (1997), IEEE Computer Society, pp. 141–151.

[13] Rivest, R. L., and Shamir, A. Payword and micromint: Two simple micropayment schemes. In *Security Protocols Workshop* (1996), T. M. A. Lomas, Ed., vol. 1189 of *Lecture Notes in Computer Science*, Springer, pp. 69–87.

[14] Thayer, F. J., Herzog, J. C., and Guttman, J. D. Strand spaces: Why is a security protocol correct? In *IEEE Symposium on Security and Privacy* (1998), IEEE Computer Society, pp. 160–171.

H-4010, POBox 12
Debrecen, Hungary