**PayWord**

This is my Information Security project It models a PayWord protocol, as described in docs/RS96a.prepub.pdf.

##Security architecture For this project, the SHA-1 hash function will be used; SHA-1 "digests" messages and outputs 160 bits. The messages are signed using RSA digital signature.

The identities of the Broker, User and Vendor are their email addresses, represented on 1024 bits; Each key has a length of 1024 bits. The expire date is stored as a long value, as given by Java's Date class, therefore another 64 bits. The other info: account no, credit limit, etc. will be stored as long values, each of 64 bits.

The steps to be taken: ###1. Registration: A. User sends to Broker: his identity, his public key, his account no and credit limit. B. Broker sends to User: the certificate: C(U) = sigB(B, U, KB, KU, E, I), where B is the identity of the Broker, U is the identity of the User, KB is the public key of the Broker, KU is the public key of the User E is the expire date of the certificate, I are the additional information: account no, credit limit.

###2. Payment: If the User makes his first payment to the Vendor: User generates a hash chain of length n: generates cn, an random secret (number), then c(n-1) = h(cn) and so on until c0 = h(c1) User computes a commitment: commit(V) = sigU(V, C(U), c0, D, I), where V is the identity of the Vendor, C(U) is the certificate of the User, generated by the Broker, c0 is the root of the hash chain, D is the current date, I are additional info: length of the chain, etc. User sends the commit to the Vendor. User sends the i-th payment of the day to the Vendor by sending the pair (ci, i, value). ###3. Redeem: Considering (cl, l) the last payment pair, the Vendor send to the Broker a message containing: commit(U), cl, l.

##Client-Server architecture ###Broker: The Broker will act as a Server, waiting for connections and requests from User(s) and Vendor(s).

###User: The User will act as a Client, it will start connections with the Broker and Vendor(s).

###Vendor The Vendor will act as both a Client and a Server, it will wait connections from the User(s) and start connections with the Broker.

|  |
| --- |
| **Page 1** |

**Payword**

• electronic micropayment scheme based on hash chains

[https://people.csail.mit.edu/rivest/pubs/RS96a.prepub.pdf](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=https://people.csail.mit.edu/rivest/pubs/RS96a.prepub.pdf)

• Shopping used to create what are low (1 cent, $ 1)

• repeated intended to provide a small amount of information:

- payment for visited web pages

- pay for songs, downloaded movies

- payment for news, journal articles

• Payment for the construction mechanism: reducing operations with application

public keys by replacing them with hash operations

**Scheme description**

• Note: **sig B (m)** means concatenation **m, SIG B (h (m))**

where

**h** - SHA-1 hash function ( [https://en.wikipedia.org/wiki/Secure\_Hash\_Algorithm](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=https://en.wikipedia.org/wiki/Secure_Hash_Algorithm) )

**sig** - RSA digital signature

( [http://profs.info.uaic.ro/~cbirjoveanu/RSA Signature.pdf](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://profs.info.uaic.ro/~cbirjoveanu/RSA%2520Signature.pdf)

[https://en.wikipedia.org/wiki/RSA\_cryptosystem#Signing\_messages](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=https://en.wikipedia.org/wiki/RSA_cryptosystem%2523Signing_messages) )

• participants:

- user / buyer (U)

- seller (V)

- broker (B)

• two phases:

**1. Register U to B**

U provides personal information (identity, **K U** - its public key RSA, etc.) to

B on a private private channel.

B sends a Certificate payword **C** U **(U)**

• **B → U: C (U) = sig B (B, U, K B, K U, exp info)**

where:

B / U - Identity Broker / User

K B / K U - RSA public key of B / U

exp - expiration date

info - serial, credit limit, etc

|  |
| --- |
| **Page 2** |

- U check C (U) by checking the signature of B (using public key RSA a

of B)

- By C (U), B authorizes U to make payments to sellers using this

certified, ie to build hash chains

**2. Payment**

If U wants to make a payment to V for the first time that day, then U

generates a new hash chain

**c n**

**c n-1 = h (c n)**

**c-2 = H (c n-1)**

**:**

**c 1 = h (c 2)**

**c 0 = h (c 1)**

where

**N** - U especially suitable to permit payment sequence

**c n** - U randomly generated secret

**and c** all payword sites have the same value (for example 1c)

U computes a commitment **commit (U) U sig = (V, C (u), c 0, d, info)**

where

**V -** identity seller

**C (U)** - payword certificate of U

**c 0 -** the root of the hash chain

**d** - current date

**info** - chain; left (n), etc.

U sends the commitment to V

• **U → V: commit (U)**

- V checks U's signature on commit (U) and B's signature on C (U), the data

of expiration

- By commit (U), V is assured that U is authorized to make payments, and

all payword sites **c and** that V will receive U (until **d)** they

will redeem from B

|  |
| --- |
| **Page 3** |

U sends the first payment to V:

• **U → V c 1** 1

- verify the authenticity of the payment V: if **H (c 1) = c 0** accepting payment, otherwise

...

In the same manner, U send to take payment at V:

• **U → V: c** **i,** i

At the end of the day, V will redeem from B the received paywords from U

• **V → B: commit (U), c** **l,** l

where

**c l,** l - the last pay received by V U

- B check commit (U), the last payment (through the applications of the function h)

- if the payment is genuine and has not been redeemed in a session

then B transfers cents from U's account to V's account

**Exercise:**

Implement the Payword microplate scheme described above using

client / server communications to simulate the transmission of messages between

participants. Implementation must allow payments for value products

different and check the cases where U is trying to use the same paywords

to buy more products from V.