Using Trust Management to Support MicroPayments

Simon N. Foley, Thomas B. Quillinan
Department of Computer Science,
University College, Cork, Ireland.
{s.foley,t.quillinan}@cs.ucc.ie

Abstract

This paper illustrates how hash-chain based micropayment schemes can be codified within the KeyNote Trust Management system. We describe an application of this scheme to the secure WebCom system, a distributed computation architecture. This supports micropayments made by servers to clients for computations performed and, payments by clients to servers for services.

1. Introduction

A metacomputer [2, 10] is a network of coordinated computers that work together to solve large problems. The metacomputer provides an architecture that manages the distribution of resources and computation across individual systems that range from user workstations to specialized servers. This approach encourages the sharing of resources between different organizations and individuals. A consequence of this sharing is that the providers of resources expect payment for their computation processing and services. For example, participants might be paid for contributing processing power to weather model computation.

This paper describes a payment mechanism for the WebCom metacomputer [10] under development at University College Cork. WebCom provides architectural support to servers scheduling mobile computation and services to client systems. KeyNote [3] provides trust management for Secure WebCom [8]: server credentials prove authorization (to clients) to schedule certain services; client credentials prove (to servers) the right to receive certain services. This paper illustrates how the KeyNote based authorization mechanism in Secure WebCom can be used to support hash-chain based micropayment schemes such as [1, 6, 11].

The paper is organized as follows. Section 2 provides an overview of the principles of hash-chain based micropayment mechanisms. While existing hash-chain based micropayment schemes use dedicated mechanisms, Section 3 illustrates how a trust management system can be used to implement these mechanisms. The advantage of this approach is that payment requirements and processing may be expressed as a component of a wider authorization policy. This approach has been used to integrate micropayments in WebCom as described in Section 4.

2. Hash Chain based MicroPayments

Micropayments schemes are intended to support very low-value payments and operate as follows.

A *payer* (the principal making the payment) generates a fresh random seed *s*, and computes $h^n(s)$, where *h*() is a cryptographic one-way hash function. If *s* is known only to the payer, then $[h^{n-1}(s),n\text{-}1,val]...[h^1(s),1,val]$ provides an ordered chain of micropayments, each one worth *val*. Initially, the payer provides a *payee* with $[h^n(s),n,val]$, which acts as a contract for (*n*-1) micropayments.

A payee (the principal receiving the payment) who has securely received *i* micropayments, $[h^{n-1}(s),n\text{-}1,val]...[h^{n-i}(s),n\text{-}i,val]$, can use the hash function *h*() to check their validity against the initial contract. Since *h*() is a one-way hash function it is not feasible for the payee to forge or compute the next $(i+1)^{th}$ payment (before it is paid). Micropayments may be cashed in at any time; the payer keeps track of contracts issued to guard against double spending.

This approach to micropayments has been proposed and used in payment schemes proposed by [1, 6, 11]. For example in [1] the payer threads digital coins (issued by a bank) through the hash chain such that each micropayment reveals an authentic digital coin that can be reimbursed by the original bank.

3. MicroPayments in KeyNote

Trust Management [3, 4, 12] is an approach to constructing and interpreting the trust relationships between public keys that are used to mediate security critical actions. Cryptographic credentials are used to specify delegation of authorization for services among public keys.

We use the KeyNote trust management system [3] to provide support for micropayment authorization of services between public keys across networks. We use a simple example to illustrate our approach.

Alice (public key **`kAlice`**) expects payments for providing services X and Y. She trusts banks Bank1 or Bank2 to guarantee payments for the customers of her services up to a certain limit (EUR50.00). This is expressed by the KeyNote policy credential in Figure 1.

```
Authorizer: "POLICY"
Licensees:  "kBank1" || "kBank2"
Conditions: @Val * Num <= 50.0 &&
        Service=="X" || Service=="Y";
```

Figure 1: Alice's Policy

This *policy* credential defines the conditions under which the public keys **`kBank1`** and **`kBank2`** are trusted by Alice. These conditions are defined using a C-like expression syntax in terms of attributes Val (micropayment value *val*), Num (number of micropayments *n* in contract) and Service.

Figure 2 provides a micropayment contract credential that a customer (public key **`kCust`**) buys from Bank1. It is signed by the owner of public key **`kBank1`**, delegating authority over the contract to the customer.

```
Authorizer: "kBank1"
Licensees:  "kCust"
Conditions: @Val==0.01 && Num=1000 &&
   Contract == "tIyJelErKyLtYoIJQPw/bQ==";
Signature: ....
```

Figure 2: Customers Contract Credential

Attribute Contract provides the initial contract value $h^n(s)$ where s is the secret seed known only to the bank and customer. This particular contract is for a maximum of 1,000 micropayments valued at EUR0.01 each. Note that the bank could alternatively decide to write a different credential that delegates authority to generate contracts directly to the customer.

When requesting service X, the customer provides $[h^{(1000-i)}(s),...]$ as the i^{th} micropayment to Alice; in this case we assume that **kCust** has already made payments $[h^{999}(s),...]$ to $[h^{(999-i)}(s),...]$ to Alice. Alice uses the trust management system to determine, given the credentials, whether the customer is authorized for the particular request. Attribute bindings [Val=0.001; Num=1000; Service=X; Contract=$h^i(v)$] define the circumstances of the request. Alice uses the KeyNote query engine to search for a delegation chain that links the trusted key **kBank1** (from her policy credential) to **kCust** (the requester) and satisfies the circumstances of the request.

By casting micropayments in terms of trust management one obtains a framework that provides flexibility in managing complex payment and service authorization trust relationships. For example, Alice might decide to out-source her support for service X to Bob by writing an appropriate KeyNote credential that delegates her authority (contract with the customer) to Bob.

4. MicroPayments in Secure WebCom

We are exploring how this approach can be used to manage payments for services within the WebCom meta-computer developed at UCC. WebCom [7, 10] provides architectural support for servers scheduling mobile computation to clients. KeyNote provides trust management: server credentials prove authorization (to clients) to schedule certain services; client credentials prove (to servers) the right to receive certain services. By encoding payment contracts within these credentials, servers can charge clients (and vice-versa) for the services that they provide.

We have found that our KeyNote micropayment based approach is well suited to the WebCom metacomputer architecture which may be used for carrying out distributed computation tasks across networks of workstations. The initial overhead incurred by a WebCom server when computing a micropayment contract for a client is offset by the subsequent computational burden that is off loaded (and paid for, task by task) to the client. Client side checking of payments carries minimal overhead, becoming part of the existing credential based authorization check.

Blaze et. al. [5,14] use KeyNote to manage trust for a microbilling based payment scheme. Their scheme is similar to IBM's minipay scheme [9]: KeyNote is used by the payer (merchant) to determine whether or not an offline payment from a particular payee (customer) should be trusted, or whether the payee should go online to validate the payment and payee. The scheme is intended for small value payments (under $1.00). Since each payment transaction requires a public key cryptographic operation it may not be practical

for very small payments where the cost of processing is high relative to the value of the payment.

5. Discussion and Conclusion

In [13], Shirkly argues that micropayments struggle to attain widespread usage because of their very nature. He states that users require "predictable and simple pricing" whereas micropayments "waste the user's mental effort in order to conserve cheap resources". When a user interacts with a Micropayment system, they still have to make a choice – whether or not X is worth Y. For example, imagine a website where a micropayment is required for each page viewed. A user will not automatically view each page without carefully considering if that webpage is worth spending a tiny amount of money for. The amounts being considered are so small that this decision is wasteful. The only case where a user is willing to accept a transaction automatically is where that transaction is free.

In the system proposed in this paper, this is not an issue. Users do not make decisions on which transactions to accept – instead the decision to accept (trust) a transaction is performed by KeyNote. In effect the user's mental effort is represented in terms of a KeyNote policy.

Another critism of micropayment schemes is based on the ratio of setup cost versus the total cost of the transactions. Ideally the relatively large cost of initialisation is offset by the frequent use of the low cost payment over time. With the application described in this paper, this is particularly true. Relationships between payer and payee are generally long term and this setup cost is insignificant compared to the total cost of the transactions processed.

In this paper we have shown a new scheme for micropayments using the KeyNote trust management system. This scheme has been implemented in the WebCom Meta-computer, providing a method of rewarding clients for work completed.

Our proposed scheme can be extended to provide for an evolution of trust between the server and client. The client originally does not trust the server to reimburse the micropayment, and might seek to (inefficiently) cash in each micropayment as it arrives. After *n* such reimbursements have been made the client might write a new KeyNote policy stating that it is safe to seek reimbursement after every *m* micropayments.

Acknowledgments

The support of the Informatics Research Initiative of Enterprise Ireland is gratefully acknowledged.

References

[1] Ross Anderson, Harry Manifavas, and Chris Sutherland. Netcard - a practical electronic cash system. In *Cambridge Workshop on Security Protocols*, 1995.

[2] A. Baratloo, M. Karul, Z. Kedem, and P. Wyckoff. Charlotte: metacomputing on the web. In *9th International Conference on Parallel and Distributed Computing Systems*, 1996.

[3] M Blaze et al. The keynote trust-management system version 2. September 1999. Internet Request For Comments 2704.

[4] M Blaze, J Feigenbaum, and J Lacy. Decentralized trust management. In *Proceedings of the Symposium on Security and Privacy*. IEEE Computer Society Press, 1996.

[5] Matt Blaze, John Ioannidis, and Angelos D. Keromytis. Offline micropayments without trusted hardware. In *Financial Cryptography*, Grand Cayman, February 2001.
[6] Jean-Paul Boly et al. The ESPRIT project CAFE - high security digital payment systems. In *ESORICS*, pages 217–230, 1994.
[7] Simon N. Foley, Thomas B. Quillinan, and John P. Morrison. Secure component distribution using webcom. In *Proceeding of the 17th International Conference on Information Security (IFIP/SEC 2002)*, Cairo, Egypt, May 2002.
[8] S.N. Foley, T.B. Quillinan, J.P. Morrison, D.A. Power, and J.J. Kennedy. Exploiting KeyNote in WebCom: Architecture neutral glue for trust management. In *Fifth Nordic Workshop on Secure IT Systems*, Reykjavik, Iceland, Oct 2001.
[9] A. Herzherg and H. Yochai. Mini-pay: Charging per click on the web. In *Sixth International World Wide Web Conference*, 1997.
[10] John P. Morrison, David A. Power, and James J. Kennedy. WebCom: A Web Based Distributed Computation Platform. Proceedings of Distributed computing on the Web, Rostock, Germany, June 21 - 23, 1999.
[11] Torben P. Pedersen. Electronic payments of small amounts. In *Security Protocols Workshop*, pages 59–68, 1996.
[12] R Rivest and B Lampson. SDSI - a simple distributed security infrastructure. In *DIMACS Workshop on Trust Management in Networks*, 1996.
[13] C. Shirky. The Case Against Micropayments. In http://www.openp2p.com/pub/a/p2p/2000/12/19/micropayments.html
[14] J. Ioannidis, S. Ioannidis, A. Keromytis and V. Prevelakis: Fileteller: Paying and getting paid for file storage. In *Proceedings of Financial Cryptography*, March 2003.