SECURE GRID-BASED MULTI-PARTY MICROPAYMENT SYSTEM IN 4G NETWORKS

Manel Abdelkader[1], Noureddine Boudriga[1] and Mohammad S. Obaidat[2]
[1]*CN&S Res. Lab., University of November 7th at Carthage, Tunisia*
[2]*Department of Computer Science, Monmouth University*
W. Long Branch, NJ 07764, USA

Keywords: Micropayment, 4G networks, grid architecture, delegation, tracing payment, Network-based/Grid Computing.

Abstract: Grids present an attractive type of distributed applications that can be efficiently developed on 4G networks. They are characterized by large scale resource sharing and innovative distributed applications. The design and deployment of a service in a 4G Grid platform can be done in real time without a prior knowledge of any contributing node. It can be used efficiently to implement sophisticated applications while providing a complete control. In this paper, we propose a new secure micropayment scheme based on the Grid concept. Our scheme presents a solution to pay anonymous parties present on various 4G networks, while allowing tracing payment operations.

1 INTRODUCTION

In the economic world, payment presents one of the main mechanisms motivating individuals and communities to share their goods. In general, payment is based on exchanging amounts of payment means for some required services or goods. This exchange should be protected against the misbehavior of the customer and the buyer as well as against any external threat. For this purpose, mechanisms with different levels of security are employed according to the value of the transactions. Further, trust third parties are defined to control payment between involved entities (Gu et al., 2004), (Buyya et al., 2005), (Buyya and Vazhkudai, 2001), (Barmouta and Buyya, 2003), (Ho and Huang,1990), (Crispo, 2001), (Rivest and Shamir, 1996), (Manasse, 1995), (Micali and Rivest, 2002), (Yang and Garcia-Molina, 2003), (Holzmann and Gerard, 1988), (Obaidat and Boudriga, 2007).

Nowadays, the majority of business transactions are conveyed to the electronic world. Such evolution induces the need to define economic models suitable to the nature of the new world. The first proposed mechanisms were based on macropayment (Barmouta and Buyya, 2003). The latter is characterized by the transfer of important amount of electronic money on the networks. These mechanisms do not allow a fine management of payment when accessing a service. In addition, the conclusion of transactions with high values requires the establishment of strong security mechanisms and on-line verification systems.

To respond to these requirements and refine payments according to the nature of the offered services, a second type of payment was defined through micropayment (Rivest and Shamir, 1996), (Manasse, 1995), (Micali and Rivest, 2002), (Yang and Garcia-Molina, 2003), (Obaidat and Boudriga, 2007). The latter is based on small payment values management. For this kind of payment, even if some losses are tolerated, security remains a serious concern. In fact, for both types of micropayment, anonymous or related to the payer, security should be guaranteed. For anonymous micropayment, there is no relation between the payer and the payment means or coins. In this case, the coins should be protected by a third party which is in general a bank. The latter should guarantee the integrity and the authenticity of each coin defined in the network which means also that every node wishing to verify a coin should consult the bank.

The second type of payment is related to the payer. In this case each payment mean or token should include the identity of the first payer. Thus, before accepting any payment mean, a node should authenticate the first payer and verify that he owns

Abdelkader M., Boudriga N. and S. Obaidat M. (2007).
SECURE GRID-BASED MULTI-PARTY MICROPAYMENT SYSTEM IN 4G NETWORKS.
In *Proceedings of the Second International Conference on e-Business*, pages 137-148
DOI: 10.5220/0002115801370148

the value of each payment mean. This verification requires the involvement of a trusted third party. In addition, the payee can directly redeem the payment means or use the same token for another payment, if the micropayment mechanism allows asking for a delegation authorization. In this case, every payee in the network should verify the chain followed by the payment mean since it has to be spent by the first payer (Obaidat and Boudriga, 2007).

Consequently, micropayment still requires the definition of appropriate security measures, which could become complicated according to the number of the payers and the nature of the payment means and payment chains. Further, it does not define mechanisms allowing to conclude distributed payment or pay distributed applications. This kind of applications is widely needed in the 4G networks, which are characterized by the inter-operability of different heterogeneous access networks composing different types of networks with diverse underlying protocols. Therefore, when accessing a service provided on 4G networks, a node can be served simultaneously by various service providers belonging to different networks. Further, resources may vary dynamically during service provision according to the requestor's node and the service provider's mobility. The study of these issues becomes more interesting when we know that a node can not identify all the resources contributing to service provision. Thus, all these factors should be taken into account during the design of payment protocols. A significant example of 4G distributed applications can be built through the study of the characteristics of GRIDs.

Grids present an attractive area of application characterized by large scale resource sharing and innovative distributed applications. They enable the sharing and coordinated use of resources in dynamic collaborations. Resource sharing is not limited to file exchange; it can provide on-demand access to all kinds of computational resources. For Grids, the sharing of resources is highly controlled. In fact, resource providers and consumers need to negotiate in real time resource sharing arrangements including the nature, the security and the policies of the share. Thus, GRID presents an interesting dynamic architecture. In fact, the construction of a service is done in real time without a prior knowledge of any contributing node. Further, the first requester ignores the manner with which his request is handled. However, a network administrator can retrace the service architecture. These features could be found in micropayment and thus they would be used in our system.

In this paper, we propose a secure micropayment scheme based on the Grid paradigm. Our scheme takes into consideration the nature of the distributed application and resources sharing. In fact, it defines mechanisms allowing to freely manage micropayment means at different nodes of the network. Thus, a consumer may allow providers to re-assign new values and re-use micropayment means for other purposes. In addition, in Grid environment, a consumer does not need to have knowledge about resource providers or service architecture. Consequently he cannot identify to whom he should pay. In our scheme, we also present a solution to pay unknown parties without using anonymous means since we should be able to trace payment operations. Further, we use the architecture of GRID services to define a security model for micropayment, which allows protecting the involved parties in a distributed manner.

The remaining of this paper is organized as follows: Section 2 presents the main features and shortcomings of micropayment schemes. Section 3 introduces the multi-party micropayment scheme and defines the generation and the distribution mechanisms. Section 4 presents the related verification and tracing mechanisms. Section 5 shows some applications of the micropayment scheme. Section 6 generalizes the micropayment scheme for other application fields. Section 7 discusses the security features of the proposed scheme. Finally, Section 8 concludes the paper.

2 MICROPAYMENT SCHEMES

In this section, we will introduce the main micropayment schemes proposed in the literature. We will focus on the advantages and the drawbacks presented by each method for payment efficiency and security. Micro-payments schemes are useful in all those scenarios where many payments of small amount of money are expected. During the mid nineties a significant amount of research has focused on developing micro-payments protocols: Millicent (Manasse, 1995), MicroMint and PayWord are among the most famous examples (Rivest and Shamir, 1996).

In recent years, a strong need for new payments proposals has given new energy to the micro payment concept. Micali and Rivest have revisited the PayWord protocol and the Rivest's Lottery approach (Micali and Rivest, 2002), solving some existing problems. In fact, one of the major problems with payments of small amounts is that the bank's processing cost can be much higher than the transferred value. The most convincing solution is to aggregate small payments in fewer larger payments. Other problems, such as the computational time

needed to perform signature operation, are no longer important as it was some years ago, because of the deployment of powerful processors and the ongoing improvement of the signature technology itself. Another important issue, in peer-to-peer applications, is that there is no clear distinction between merchants and customers: there are simply peers, which can be merchants, customers or both. In such a context, the idea of transferable coins was introduced, and PPay was one of the approaches based on it, (Yang and Garcia-Molina, 2003), (Obaidat and Boudriga, 2007).

Yang and Garcia-Molina (Yang and Garcia-Molina, 2003) proposed a protocol (PPpay) that does not involve any broker for each peer's transaction. The concept of floating and self-managed currency is introduced. The payment means or coins can flow from one peer to another, and the owner of a given coin manages the currency itself, except when it is created or cashed, which means that the user manages all the security features of the owned coin(s). As other micropayments systems, PPay coin fraud is possible. PPay considers that frauds are detectable and malicious users can be punished. Moreover, it assumes that a fraud can be operated only over small amounts of money, and risk is higher than benefit.

A study of the available approaches distinguishes the following characteristics of micropayments:

- The knowledge of the path between the sender and the receiver is required. In fact, before defining the cost of packets' transmission, a node should negotiate all the charges defined by nodes present in the path. Then, the sender or the requester can choose the path with the reduced cost.
- The definition of two major means for payment is witnessed: cash or through the use of on-line connections to a third party.
- The assignment of fixed values to the micropayment tokens does not allow a fine management of payment means.
- The re-use of the whole value of a micropayment mean is authorized when delegation is possible. The payee can only manage for whom the micropayment mean will be transmitted during the next payment.

To develop a micropayment scheme, different conditions should be fulfilled. First, efficiency should be guaranteed. In fact, the cost of the communication and processing related to micropayments should be kept as low as possible; otherwise, it may exceed the value of the payment itself. The importance of this feature should not greatly affect other properties related to security and fairness. Second, a micropayment system should protect the rights of payers and payees. For this purpose, security mechanisms should be included in this system. Among the main security threats against which a micropayment system should be protected, we can mention the double spending, the forgery and theft of coins. Third, a micropayment system should be scalable and flexible. It should support the augmentation of the number of transactions and be independent from the nature of payment.

Even if the schemes proposed for micropayment may differ, common features can be defined. Practically, each scheme should define a technique for money generation and money redemption. In addition, it should propose techniques for payment verification. Other features should be present for distributed applications related to dynamic and flexible management and payment traceability.

3 GRID BASED MICROPAYMENT SYSTEM

In this section, we introduce a novel micropayment system. The contribution of our work is the study of micropayment based on GRID application concept. In fact, in most existing approaches, the payment should be done through predefined accounting system in which users are defined by accounts.

In our approach we will foresee the case where a customer will pay the first service provider he knows. The latter will spend the received tokens without the need for redeeming at the broker. The next node that receives those tokens can use them and so on. The tokens continue to be spent on the network until a node remarks that the token will expire, then it will redeem it at the broker. The latter contains the accounts of the customer and the service providers.

Another new feature introduced in our work is related to coins distribution. In fact, as far as we know, there is no proposition that allows the subdivision of the value of a micropayment token into smaller values assigned to other tokens. In this paper, we present the subdivision procedure which is performed by the involved nodes without returning to the broker. In the following subsections, we present the micropayment algorithm and the related tracing process. Three main actors are defined in our scheme; they are: a) the customer ***C,*** who is identified by ID_C and defined as the service requestor and payer; b) the broker B, who is identified by ID_B and is responsible for the generation and the protection of the micropayment tokens; and c) the service providers P_i, $1 \le i \le n$,

identified by ID_{Pi}. In fact, service providers, P_i, should be able to re-use the micropayment tokens without getting back to the broker.

The micropayment algorithm is depicted as follows.

Micropayment Algorithm

1. *The consumer generates an unbalanced one way binary tree UOBT, as presented in (Ho and Huang,1990). For this purpose, C chooses a random value A_{NT}, two integers N and T, and two hash functions H_1 and H_2. A_{NT} denotes the value associated to the tree root.*
   C starts by applying H_1 N times to A_{NT}. This operation results in the construction of a backbone hash chain (A_{NT}, $A_{(N-1)T}$,…, A_{1T}) where $A_{kT}=H_1(A_{(k+1)T})$ for $1\leq k<N$.
   Each A_{kT} with $1\leq k\leq N$ forms a secret root of a sub-chain derived from the application of the second hash chain H_2. In fact, for a given A_{kT}, $1\leq k\leq N$, C applies H_2 T times.
   Then, the resulted sub-chains (A_{kT}, $A_{k(T-1)}$,…,A_{k1}) where $A_{kt}=H_2(A_{k(t+1)})$ for $0\leq t<T$ are defined.
   After the generation of all the sub-chains related to the backbone hash chain, C defines the anchor vector $A=(A_{10}, A_{20},…,A_{N0})$ where each A_{k0}, $1\leq k\leq N$ is the anchor value of a sub-section.
2. *The user C forwards a signed message that consists of the anchor vector $A=(A_{k0})_{1\leq k\leq N}$ with the length of the sub-chains and the value assigned for each anchor value, to obtain a broker commitment. Let $V=(v_i)_{1\leq i\leq N}$ be the vector containing the values corresponding to the anchor vector.*
   Then, a token request, defined by $\{ID_C, A, V\}$, is signed by C and sent to B, while keeping A_{NT} secret.
3. *B generates a signed commitment corresponding to each anchor value in A (The procedure followed by B in this step will be detailed in the sequel). Then, B sends the commitments to C.*
4. *C requires the total cost for accessing a service offered by a provider P_i. According to this cost, C selects the suitable commitment to be used for micropayment tokens generation.*
5. *C and P_i sign a contract defining the parameters of the micropayment.*
6. *C generates micropayment tokens and sends them to P_i according to the terms of the signed contract.*
7. *P_i verifies the integrity of the received tokens. Verification is done off-line (without returning to B).*
8. *If the verification result is positive, then P_i proceeds in one of the following three different manners:*
   a) *Send the token to B to be redeemed. The micropayment process will then end.*
   b) *Reassign the same token and re-use it for other purposes. In this case, P_i asks C to allow him to re-use the token. For this purpose, a delegation procedure is followed and C states that a specific token can be freely used by P_i.*
   c) *Subdivide the value of the token to other smaller values defined in different tokens. In this case, P_i will ask C to allow him the subdivision of a token into "sub-tokens" which could be used for different and independent purposes.*

In the following we will detail the generation and distribution processes introduced by the micropayment algorithm.

3.1 Generation Process

The generation process is composed of two basic steps: (i) the generation of a generic proof of possession of a global amount of money. This step is performed between *C* and *B;* and (ii) the generation of micropayment tokens, which are generated by *C* to pay services offered by provider P_i.

Step (1): Customer C has an account at the broker ***B***. When ***C*** spends an amount of electronic money, the distributed sum should be reduced from his account. To prove electronic money possession, *C* requires a commitment signed by the broker for each element present in the anchor vector *A*. A signed request sent by *C* should contain:

$$\{ID_C, A, V\}_{signed\ by\ C},$$

On requesting a commitment, *C* authorizes the broker to block the amount of money present in the commitment. The amount is defined by the value of every element of vector *V* and is assigned to an anchor value in *A*. For each anchor value of the vector, *B* generates the corresponding commitment. Let $G=\{G_i, 1\leq i\leq N\}$ denotes the set of the generated commitments and let commitment G_i be defined by:

$$G_i = \{ID_C, ID_B, s_i, A_{i0}, v_i\}_{signed\ by\ B}$$

where s_i is a unique serial number allowing to identify each commitment at B and assigning the generated tokens to the suitable commitment. Then ***B*** sends the signed commitments to ***C***.

Step (2): Token generation depends on the payment policy defined by each service provider. Among many clauses, the policy indicates the approved procedures with which a given service could be paid. These procedures define a set of recognized distribution functions $\{f_d, 1\leq d\leq D\}$ defined as the functions allowing the subdivision of the global cost into different sub-values paid consecutively. These functions vary according to the nature and the demand of the required resource.

When ***C*** contacts the first service provider P_1, he will ask for the global cost *K* required for service provision, the distribution functions

required $(f_d, 1 \le d \le D)$, the nature of the tokens (purchasable or distributable) and the value k_1 of the first token. Upon receiving a response, ***C*** chooses the commitment G_i with which he will make the payment. Then, he presents a first version of a contract to P_1 under the following form:

$$\overline{C}_{C-P} = \{ID_C, ID_{P_1}, G_i, f_d, d\}_{signed\ by\ C},$$

where $d \in \{0,1\}$. If $d = 0$, then ***C*** will generate purchasable tokens (i.e., tokens that can be only changed into currency at B). If not, ***C*** will generate distributable tokens (i.e., tokens that could be used by P_1 for other payments). Then, P_1 verifies the signature of the broker on G_i and the amount assigned to A_{i0} $(v_i \ge K)$.

If the result of the verification is negative, P_1 asks for other commitments. With a positive result, P_1 signs $\overline{C}_{C-P}$ to construct the final contract:

$$C_{C-P} = \{(ID_C, ID_{P_1}, G_i, f_d, d)_{signed\ by\ C}\}_{signed\ by\ P_1}$$

Then, it sends it back to *C*.

According to the chosen distribution functions, *C* can consider the set of redeemable values $\{k_j, 1 \le j \le n\}$, where k_j is defined by $k_j = f_d(A_{ij})$. We note that knowing f_d and the first value k_1, one can deduce the number *n* of tokens that should be delivered from *C* to P_1. Consequently, *C*, *B* and P_1 know that *n* elements of the sub-chain corresponding to $\boldsymbol{G_i}$ would be used as well as the value assigned to each element.

C generates and sends to P_1 the micropayment tokens during service provision. A purchasable token has the following form:

$$t_j = \{ID_C, ID_{P_1}, s_i, A_{i(j-1)}\}.$$

While a distributable token has the form:

$$t'_j = \{ID_C, ID_{P_1}, s_i, A_{i(j-1)}, k_j, d\}_{signed\ by\ C}.$$

where *d* states that ***C*** delegates the right of token distribution to P_1. Thus, P_1 is not able to change t'_j into currency.

3.2 Distribution Process

Distribution is based on the accountability delegation protocol presented in (Crispo, 2001), assuming that a delegator can transfer accountability when he transfers his own rights to the delegated node. It has been formally shown that accountability is provable to third parties, even if the given property has been transferred by means of delegation. Using accountability delegation protocol in our micro-payment scheme means that:

- When *C* authorizes a service provider *P* to distribute a token, he also transfers the right of distributing it to another peer or changing the token into currency.
- When *C* transfers to *P* the right of distributing or changing the token into currency, he also transfers to *P* the accountability for exercising such a right. In other words, if *P* commits a fraud with the distributed token, the link between *P* and his actions could be established.
- If *C* is not the valid owner of the given token, *P* can refute distribution after the verification of the token validity.

Now, we describe the case where *C* assigns a token to P_1 and allows him to distribute it. The delegation procedure begins when *C* asks provider P_1 about the nature of the token he wants. When P_1 chooses distributable tokens, *C* generates and signs an authorization token. An authorization token includes information about ***C***, P_1, and the transfer of the accountability. Upon authorizing P_1 to distribute a token, ***C*** becomes no longer responsible of this token. The authorization token serves as a proof at the broker and the other peers to whom P_1 will distribute the related micropayment token. We define an authorization token as:

$$\theta = \{ID_C, ID_{P_1}, s_i, A_{ij}, f_d, k_1\}_{signed\ by\ C}.$$

After receiving the two tokens (micropayment and distribution authorization), P_1 could make other micropayment based on *C*'s micropayment token. In fact, from a micropayment token, P_1 could derive m tokens such that:

$$t_{j,1} = \{ID_C, ID_{R_1}, s_i, A_{i(j-1)}, 1, k'_{j,1}, \theta, d_1\}_{P_1}$$

$$t_{j,1} = \{ID_C, ID_{R2}, s_i, A_{i(j-1)}, 2, k'_{j,2}, \theta, d_2\}_{P_1}$$

…..

$$t_{j,m} = \{ID_C, ID_{R_m}, s_i, A_{i(j-1)}, m, k'_{j,m}, \theta, d_m\}_{P_1}$$

with ID_{R_i} is the identifier of the resource, θ is the authorization token and $\sum_{j=1}^{m} k'_{i,j} \le k_i$.

Re-assignment presents a special case of distribution. It refers to the transfer of the ownership

of a micropayment token to another entity. In this transfer, the token keeps the same value and the new owner cannot modify it. In this case, we consider that the distribution function chosen by the entity who receives the distributed token is constant. Therefore, the whole value of the token is transferred to the same entity.

Figure 1 illustrates the use of tokens for micropayment. In fact, ***C*** uses three different UOBT sub-chains: A_1 to pay P_1, A_2 to pay P_2 and A_3 to pay P_3. C generates purchasable tokens for P_3. The latter can only redeem those tokens at the broker. However, C generates distributable tokens for P_1 and P_2. Thus, P_1 derives other micropayment tokens and use them to pay $P_{1,1}$ and $P_{1,2}$. We show also in the figure that P_2 can receive a distributed token from $P_{1,1}$ derived from a micropayment token originating from C.

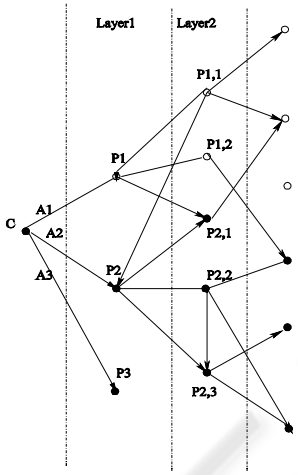

Figure 1: Distributed micropayment scheme.

4 MICROPAYMENT VERIFICATION AND TRACING

In this section, we define the verification procedure followed by a node receiving a micropayment token. Then, we present a tracing method allowing the reconstruction of a micropayment operation.

4.1 Verification Procedure

Verification procedure depends on the nature of the received token. We distinguish two types of verification.

Purchasable tokens: The first type is related to purchasable tokens. The verification procedure involves three steps.

First, P_1 verifies the first version of the contract. This implies the verification of the signature of the broker on the commitment and the assigned value, which should cover the cost required for service provision. A positive verification is concluded by the signature of the final version of the contract.

Second, P_1 verifies tokens as long as he provides the service. In fact, each micropayment token is accompanied with a verification procedure, which is related to the hash value received in each token. Therefore, for the i^{th} token, the receiver verifies that $A_{i(j-1)} = H_2(A_{ij})$.

The chain of hash values proves the correctness of the payment. In the case where a token is lost or does not reach the provider, the provider should notify customer *C*. Then, *C* drops the value assigned to the lost token and affects it to the next token. Thus, the set of the tokens becomes (A_{ik}) where k belongs to $[1,j[\cup]j,n+1]$, j refers to the lost token and $k_j = f_d^{j+1}\left(A_{i(j-1)}\right)$. In addition, *C* notifies the broker about the loss to be considered during redeeming.

Third, B verifies the tokens before purchasing. In fact, to be purchased P_1 only presents the last received token with the contract to *B*, who verifies the authenticity of the contract and gives the total sum corresponding to the addition of the values assigned to the different tokens given to P_1.

Distributed tokens: In this case, the receiver should verify, in addition to the presented commitment and the contract, the authorization and the payment tokens. In this paragraph, we reach the two layers of the distribution procedure. As depicted in Figure 2, the nodes participating in the verification procedure are encircled. We consider customer *C*, the first service provider P_1 who is present in the layer 1 of the distribution architecture and the service providers $P_{1,t}$ present in layer 2 receiving distributed tokens. Verification proceeds as follows:

First, P_1 verifies the micropayment token. This includes the verification of the chain of hash values as it was previously shown, the verification of the assigned value and the validation of *C's* signature. In addition, P_1 verifies the authorization token related to this micropayment token.

Second, every $P_{1,t}$ checks the distributed tokens that he receives from P_1. The verification is performed as follows. First, verification is done between layer 2 and layer 1 of the distribution architecture. $P_{1,t}$ verifies every distributed token and the authorization token given by C to P_1.

In the case where the check succeeds, the verification becomes between layer 2 and *C*. $P_{1,t}$ sends the received distributed tokens to *C*, who verifies that the sum of the distributed tokens values does not exceed the value of the original token. For this purpose, *C* maintains a counter for each distributable token. Each time a $P_{1,t}$ sends a distributed token, *C* reduces the value of that token from the original value; and so on, until all the value of the token is spent. If a token arrives after the value has been spent, C rejects the token and informs its source that any process where the token is involved should be stopped.

This procedure is followed in the same manner between layers n and *n-2, for all n*, in the distribution architecture, as depicted in Figure 2, where, following this process, we can reconstruct the nodes participating in each verification procedure. Thus, the set containing {C, P_1, $P_{1,1}$, $P_{1,2}$, $P_{2,1}$} is defined for the verification of the distributed tokens generated by C and paid to P_1. However, the set {C, P_3} refers to the verification of purchasable tokens paid to P_3.

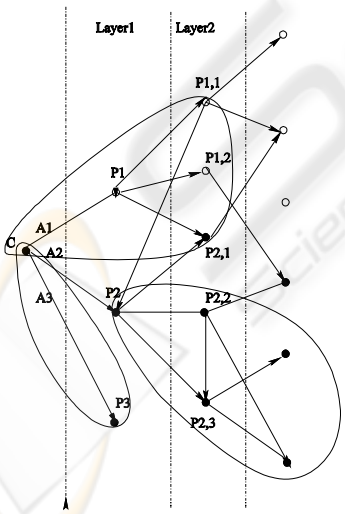

Figure 2: Verification Procedure.

Therefore, to ensure verification, we have introduced a visibility of two layers for the different nodes receiving distributed tokens. Visibility guarantees that a node present at layer *n-1* cannot spend more than the amount assigned to a distributable token. Furthermore, verification should be concluded by B before purchasing a distributed token. For this, B verifies the authorization and the distributed token. Then he redeems its value. Thus, authorization tokens serve as a proof in case of dispute or need for tracing. For this purpose, they should be kept by the network nodes involved in token distribution.

4.2 Tracing Micropayment

We present here a tracing method that allows retrieving the different tokens distributed by a consumer *C*. Distribution and re-assignment are used to resolve dispute and detect attacks. To execute a job in the GRID context, a node launches the request in the network and waits for a suitable response. A response should contain the resource able to execute the job, the *QoS* required by the requester, and the amount of money needed to accomplish the job. In a traditional GRID, there are predefined proxies in each network. When a node wants to access to the GRID, it allows the proxy to act on its behalf when searching or using resources, since the GRID has a prior knowledge of service provision in the network.

In our micropayment scheme, every node performing distribution is considered as a proxy. To reconstruct the distribution architecture, B should identify all nodes that have performed a distribution process. A broker or an auditor starts to reconstruct the distribution architecture from a received token mentioning the first originator of the token and the related anchor. From the signature present on the token, the broker deduces the sender. He asks then for the authorization token, the distribution function and the original token. The broker verifies the distribution procedure followed by the token. Then, he continues by verifying whether the token is distributed by the first customer or by an intermediate entity. In the second case, he asks for the distribution materials (i.e., authorization token, distribution function, and the original token) to be verified. This procedure is followed by the broker until he reaches the first entity that had generated the token. Thus, the broker can reconstruct the history of any token presented to him to be purchased or verified. However, it has not yet found the distribution architecture of a special token. To fulfill this criterion, we should require that the node making distribution should keep track of the contracts and authorization token.

5 MICROPAYMENT APPLICATION FOR GRID

In this section, we consider the application of our micropayment scheme on GRID environment defined for 4G networks. Two approaches can be addressed. In the first, micropayment can be considered as a GRID application accessible in a 4G network. In the second, the proposed mechanism can be used to pay 4G resources involved in GRID services provision. In fact, as it was presented before, our scheme allows delegating and distributing micropayment tokens in the 4G network. Both methods are useful in 4G distributed applications, since our scheme allows an entity to purchase through different providers, using the fact that a token includes the identities of the first originator and the final node possessing the token, despite access networks heterogeneity.

5.1 Micropayment for GRID

When revising the characteristic of GRID as a 4G distributed application, we can denote the existence of different common features with our micropayment scheme. In fact, a service requester has no knowledge about the nodes contributing to a GRID service provision and so about the structure of the paths to these nodes. A more complex payment computing protocol should be defined in function of resources nature. The first assumption is to use our micropayment scheme to pay GRID resources. This feature was considered by our scheme when a node transfers the accountability to another and allows him to distribute the micropayment tokens.

In addition, if we consider the GRID, we notice that resources are not allocated uniformly to all service requesters and that each resource is free to define its policies and its requirements. This feature is considered in our scheme since we have defined different distribution functions and we allow to all service providers to adopt suitable payment functions according to the nature of their resources and the offered QoS. Further, micropayment starts only when the payer and the payee agree on the distribution function and the first paid amount. For GRID, these functions allow to respond to the payment policies defined by the resources. The second assumption is to implement the micropayment scheme as a GRID application. For this purpose some analogies should be noticed:

- The first customer *C* is considered as a GRID client who will ask for a service. The service in our case is considered as micropayment.
- The first service provider P_1 is considered as the first proxy known by *C*. In this analogy, we will consider the case where P_1 asks for distributable tokens. Thus, *C* will generate authorization tokens which are considered as the proxy certificates that are used by the proxy to act on behalf of the user in the network. This point presents some difference with the GRID. In fact, whereas a simple user will wait for job execution, *C* can no longer be present; his broker will take his place.

5.2 Case Study

In this section we take the case of alternative operators exploiting the resources of the 4G networks. We show how the GRID can help such applications. For this, we need to discuss the role of an alternative operator.

The main feature characterizing an alternative operator is the absence of predefined infrastructure or private resources in 4G networks. In fact, the functions offered by these operators are based on the use of independent resources present on the heterogeneous access networks to construct a service and respond to requests.

The case of the alternative operators presents different similarities to a 4G GRID application. In fact, when a requester wants to access a service (e.g., a call establishment), he asks his network operator to act on behalf of him. From this stage, the service architecture and the functions defined in the network are hidden to the final user. The provision presented by the alternative operator is that he has not a prior knowledge of the architecture of the service or the structure of the network allowing the establishment of this service. An alternative operator should be able to discover, schedule, and allocate the needed resources.

For alternative operators, flexible and scalable payments present a great concern. A payment system should not only be able to support a variety of different payment mechanisms but it must be capable of efficiently handling payments to third-party partners as well as gathering call data to ensure that the financial relationship with the actual network owner is monitored effectively. This is where the complexity really starts to mount for companies seeking to interconnect all parties involved in a payment scheme by themselves. While payment systems can be complex when only basic voice services are involved, alternative operators need to be able to mix and match different packages of services and tariffs in creative ways. Further, services increasingly interact with the IP Multimedia Subsystem, which induces that service provision and

payment should be handled in real time. From the customer point of view, this complexity must remain safely hidden and service requests must be fulfilled as instantaneously and transparently as possible. In such environments, the multiparty micropayment protocol presents a good solution to overcome these constraints.

Let us take the case where a customer communicates with an alternative operator ASP_1 and asks for a given service. To provide such a service, ASP_1 will search (and find) the resources at ASP_2 and ASP_3. Each provider will assign the resources required by ASP_1 based on other networks provisions. For this example, ASP_1 only knows the two alternative operators, the remaining part of the service architecture is hidden to him. Consequently, ASP_1 does not know how the partition of the resource is done.

To this end, the customer generates the payment tokens and the authorization of distribution and sends them to ASP_1. Then, ASP_1 distributes the tokens to pay ASP_2 and ASP_3, who will verify distribution process with the customer. If the verification succeeds, each operator distributes the received tokens to the networks owning the real resources. At the end of this chain, each payee can communicate with the broker to redeem its tokens or spend them for other purposes.

6 GENERIC PAYMENT SCHEME

In this section, we show how we can adapt the micropayment scheme for the payment of important amounts independently of the nature of the supporting networks. This adaptation allows handling the various requirements of applications linked to 4G networks. In fact, to ensure inter-operability between the different types of 4G networks during service provision, a generic payment scheme should be defined.

In the following, we first present briefly the requirements of a payment operation. Then, we present the adequate modifications that we should introduce to our scheme to become suitable for payment purposes.

6.1 Payment Requirements

Compared to the micropayment, macropayment schemes transfer larger amounts. Consequently, these schemes require rigorous security measures. Generally, the approaches adopted by payment schemes use public key cryptography for authentication and for the protection of the privacy and the integrity of the transactions.

Besides, payment schemes need an on-line connection to the broker. In fact, before accepting any payment transaction, a service provider connects to the broker to verify the authenticity of the received payment means. The verification also allows preventing double spending. E-cash is an example of payment schemes. It requires service provider's broker and customer's broker to be on-line to verify payment transactions. In our scheme, a high level security for the payment transactions is guaranteed. In addition, we lighten the communications in the network by reducing the on-line connection to the broker.

6.2 Payment Scheme

For the payment scheme, we notice that we no longer need the use of hash values since amounts with important values would be spent for each transaction. Consequently, we only define the payment amount and introduce the re-assignment and distribution ability. The generic payment model is described as follows.

In response to the customer's request, the bank guarantees the possession of an amount of money. The difference from the micropayment approach consists in the value blocked by the bank which will be in this case more important. Also, the customer will not use the UOBT scheme which induces the modification of the structure of the first commitment. In fact, the broker signs the following new structure

$$\{ID_C, ID_B, S_i, V\}_{signed\ by\ B}.$$

After that and before making a payment, the customer will present the commitment to the service provider and require the nature of the payment. We keep in this model the ability of the service provider to choose between receiving purchasable or distributable payment means. Then, *C* and *P* sign a contract fixing the nature of the payment means and the amount to be transferred. The payment is made when *C* signs a "check" in which he indicates whether he is directly purchasable or distributable. In the latter case, *C* generates an authorization credential to be used by the payee during distribution.

The verification is done in the same way as defined for the micropayment. In addition to the verification of the paid amount and the authorization to credential, the two-layer verification process is employed. In fact, an entity receiving a sub-amount will refer to the entity who had given the

authorization to the payer. This operation allows checking that the payer possesses the paid amount. By this way, the broker is only consulted for a final redemption.

In our scheme, we have protected the payment transactions through the use of the digital signature for a simple or a distributed payment. In addition, we have introduced an autonomous verification mechanism in the distributed architecture.

6.3 Validation Scheme

Validation refers to check the micropayment protocol specification such that it will not get into protocol design errors like deadlock or unspecified receptions (Holzmann and Gerard, 1988). The validation scheme concentrates on verifying the aliveness and safety properties and correctness of the protocol specification. To verify logical consistency, a formal finite state machine (FSM) model of the protocol is constructed. The model specifies a set of asynchronous, communicating finite state machines (FSM). The individual behavior of an asynchronous FSM is defined by a finite set of local machine states and local state transitions. The system as a whole is defined, minimally, by the integration of all individual process states and the combination of all simultaneously enabled local state transitions.

In the following, we use this method to validate the micropayment protocol. For this purpose, we define the actors in the different protocol phases and present the FSM model for each one. In the micropayment protocol, we define two main phases, the initiation phase and the payment phase.

Initiation phase: During this phase two procedures are concluded. The fist is realized between the customer and the broker when generating the original commitment. For this, *C* defines two states:

S_{CO} : C has generated Anchor vectors, already sent Commitment to B, and C is now waiting for the reception of the signed commitment

S_{C1} : C is Waiting for a commitment and preparing for the following request

S_{BO} : B Waiting for anchor vectors

S_{B1} : Anchor vectors already sent by B

Figure 3 shows the finite state machine (FSM) related to the customer. In Figure 3, the circles indicate the states and the arrows show the transactions done to move from a state to another.

The second procedure is done between the customer and the service provider. During this procedure, both involved entities agree on a contract defining the payment parameters.

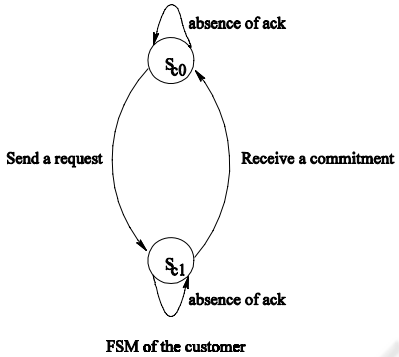

Figure 3: The customer FSM.

For this phase, the customer and the service provider define the following states.

S_{CO} = C has sent a draft of the contract to P1.

S_{C1} = C has received a final version of the contract from P1

S_{C2} = C waits for a response from P1

S_{P1} = P has received a first version of contract sent by C.

S_{P2} = P has sent a response to C

The FSM of the customer is presented in Figure 4. It shows the end of the initiation phase when arriving at state S_{C1} .

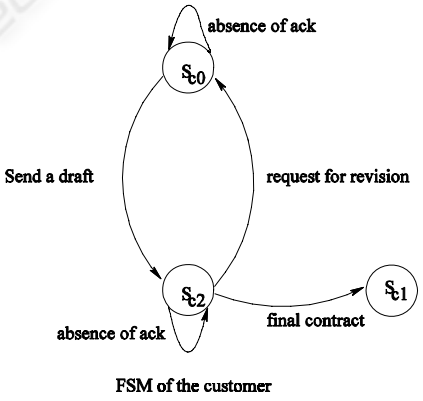

Figure 4: Customer FSM related to contract conclusion.

Payment phase. The second phase is defined for the payment procedure. In this phase, two cases are studied according to the nature of the spent tokens:

Payment without distribution property: In this case, the customer will present the suitable commitment and the related tokens. The states known by the customer are “C has sent the token, C waits for the required service”. However, the service provider defines two states “P1 has received the tokens, P1

provides service to C". For an abnormal functionality, other states appear for both participants like "payment is stopped, service provision is interrupted, or tokens are lost".

Payment with distribution property: Three actors are defined presenting three layers of the payment architecture. We denote them by P_j, P_{j+1} and P_{j+2}. The states defined for P_j are:

- P_j has sent authorization token to P_{j+1}
- P_j has sent payment token to P_{j+1}
- P_j waits distribution information from P_{j+2}
- P_j has sent distribution verification result to P_{j+2}".

For P_{j+1} we define:

- P_{j+1} has receives an authorization token from P_j
- P_{j+1} has received payment tokens from P_j
- P_{j+1} has sent distributed tokens to P_{j+2}
- P_{j+1} waits for P_{j+2} to provide him a service

For P_{j+2}, the defined states are:

- P_{j+2} receiving distributed tokens P_{j+1}
- P_{j+2} sending verification request to P_j
- P_{j+2} waiting for verification result
- P_{j+2} provides a service

The FSM models presented in the appendix detail the different transitions that the system executes when applying our micropayment protocol in the case of purchasable token and distributable token as shown in Figures 5 and 6, respectively. The two FSM models allow validating the correctness and consistency of the protocol.

7 SECURITY PROVISION

In this section, we present the security measure of the proposed scheme. In fact, we prove the how our scheme is protected against the two main attacks that threaten a payment systems, forging and double spending.

7.1 Forging Prevention

This attack is defined when a non-authorized user generates false tokens to be purchased from accounts of other entities without obtaining their approval. The proposed payment scheme prevents this attack through the definition of security measures at different levels. First, the protocol defines a commitment signed by the broker which proves that the user has a blocked amount for payment. Second, it requires the establishment of a contract signed by the payer and the payee. This contract induces that the payer and the payee approve payment transaction. Third, the protocol distinguishes between two cases. The first one consists of the use of non distributable tokens. In this case, the hash micropayment tokens are not signed by the customer. However, the use of one-way and collision-resistant hash functions protects against the generation of false tokens by other entities different from the payer.

Thus, a receiver cannot compute the antecedent of a hash value during the lifetime of a token. The second case refers to distributable tokens. The distribution process is accompanied by the signature of the original and derived tokens. In addition, the use of the authorization tokens allows verifying the legitimacy of the distribution procedure and authenticating the origin of the distribution architecture.

7.2 Double-spending Prevention

The second attack consists of the illegal use of the same token for different times and for different purposes in the network. To prevent double-spending, the protocol defines a serial number for each blocked amount present in the bank. All the delivered tokens should refer to the corresponding amount. For micropayment, our scheme adds the identifier of the hash branch from which the hash values will be distributed through the different tokens.

The second measure taken by the protocol is the use of non-anonymous model. In fact, we can notice that the different micropayment tokens include the identifier of the payer, the payee, the amount and the distribution function. The latter allows determining in advance the number and the values of the micropayment tokens at the level of the payee and at the level of the bank during redemption.

Further, the definition of authorization token and the two-layer verification for distributable tokens, allows the follow-up of tokens spending by the authorizing entity. Also, this procedure allows a payee to verify that his payer has not exceeded the authorized distributable amount and that he has not use the same value for different purposes.

8 CONCLUSION

In this paper, we have introduced the notion of multi-party micropayment system and we propose a micropayment scheme based on Grid application. Our scheme takes into consideration the nature of the distributed application and resource sharing. We also define new mechanisms to allow a tight handle of the payment means and a better management of payment operations. In addition, the proposed mechanism defines new security measures allowing real-time verification of micropayment in a distributed manner.

REFERENCES

X. Gu, K. Nahrstedt, and B. Yu, "SpiderNet: An integrated peer to peer service composition framework", 13th IEEE Int. Symp. on High-Performance Distributed Computing (HPDC-13), Hawaii, USA., 2004

R. Buyya, D. Abramson, and S. Venugopal, *"The Grid Economy"*. Special Issue on Grid Computing, M. Parashar and C. Lee (Eds). IEEE Press, New Jersey, USA, Mar 2005.

R. Buyya and S. Vazhkudai, "Compute power market: Towards a market-oriented grid", The 1st IEEE/ACM Int. Symposium on Cluster Computing and the Grid (CCGrid 2001), May 2001. Brisbane, Australia.

A. Barmouta and R. Buyya, *GridBank, "A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration"*. Proceedings of Int. Parallel and Distributed Processing Symp. (IPDPS'03). Nice, France, 2003

Yen, L. Ho and C. Huang, "*Internet Micropayment Based on Unbalanced One-way Binary Tree*", Proc. CrypTEC 99, pp.155-62, July 1999, Hong Kong.

B. Crispo. "*Delegation protocols for electronic commerce"*. In Proc. of the 6th IEEE Symp. on Computers and Communications (ISCCS01). July 2001. Hammamet, Tunisia.

Ronald L. Rivest and Adi Shamir. "Payword and micromint: Two simple micropayment schemes". CryptoBytes, vol 2(1), RSA Laboratories, 1996.

M. Manasse. *"The millicent protocols for electronic commerce"*, In Proc. 1st USENIX Workshop on Electronic Commerce, July 1995, New York, USA.

S. Micali and R.L. Rivest., "*Micropayments revisited"*. In CT-RSA, pages 149-163, 2002

B. Yang and H. Garcia-Molina, "*Ppay: micropayments for peer-to-peer systems"*. In Proc. of the 10th ACM Conf. on Computer and communication security, pp. 300-310. ACM Press, 2003, Washington, USA.

J. Holzmann, J. Gerard, *"An Improved Protocol Reachability Analysis Technique"*, Software Practice & Experience, pp. 137-161, Feb. 1988, John Wiley & Sons

M. S. Obaidat and N. Boudriga," Security of e-Systems and Computer Networks," Cambridge University Press, Cambridge, UK, 2007.

APPENDIX

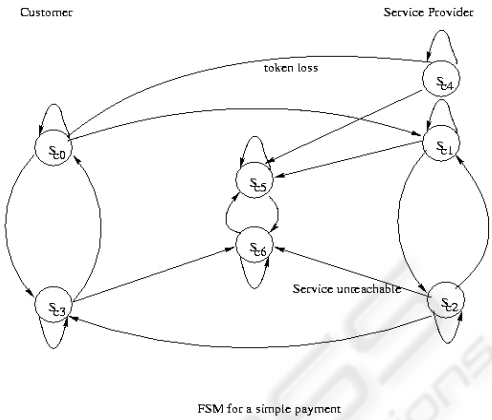

Figure 5: FSM model for purchasable token.

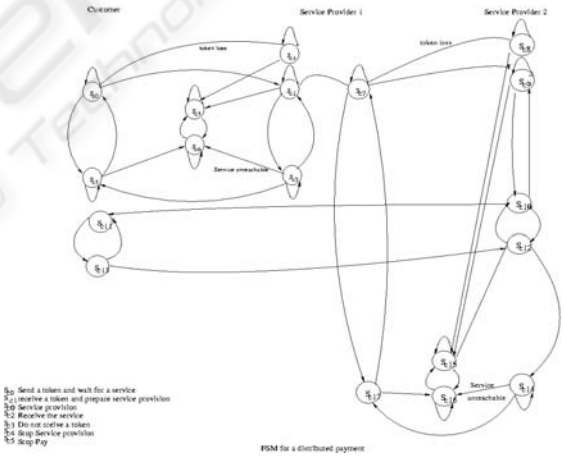

Figure 6: FSM model for distributable token.