

# On the Evidence Based Peer-to-peer Resource Management in Distributed Computing Systems<sup>1</sup>

T. C. Lam and Jyh-Charn Liu<sup>2</sup>  
brianlam@tamu.edu, liu@cs.tamu.edu  
Computer Science Department  
Texas A&M University  
College Station, TX 77843-3112

**Technical Report 2003-7-2**

**July 28, 2003**

## **Abstract**

In this paper we propose an accountability management framework for resource sharing in peer-to-peer (P2P) distributed computing systems. On the basis of a generalized e-coin paradigm, we developed the management framework based on the distribution of resource credit/policy, exchange of service and service evidences, and accountability management. A tree topology is assumed for the distribution of resource credit and policies, so that P2P parties can have an absolute reference of the credential of principals, and make the system scalable. Common resource management operations involving splitting, merging and the propagation of system attributes (such as the ownership of the resources) are bound to the resource credit to become resource evidence after the services are rendered. We propose three e-coin binding techniques for those resource delegation scenarios, so that the credit policy (distributed by the credential authority), the resource attribute (sent from the provider) and the resource order (acknowledged from the consumer) can be cryptographically tied with the warrant, the response and the challenge respectively in the e-coin for the generation of the anonymous evidence on the delegation events.

*Index terms – peer-to-peer computing, accountable resource sharing, e-coin, attribute binding.*

---

<sup>1</sup> This work is supported in part by an NSF ITR grant, EIA-0081761, and by a TAMU-CONACYT grant

<sup>2</sup> Correspondence author

## 1. INTRODUCTION

Resource management schemes can be largely divided into *centralized, hierarchical, decentralized* and *hybrid* systems [1]. A centralized management system uses one central authority to manage all resources, like the domain controller in a Windows domain. The best known hierarchical system is the Domain Name Service (DNS) [2], where the naming authority descends from the root name-servers to the lower level servers. GRID Computing [3] is a decentralized computing infrastructure to provide uniform access of shared resource. Because local domains usually have central servers to manage local resources, the hybrid scheme is used for inter-domain resource sharing.

Classical distributed computing systems are mostly based on a hierarchical administrative architecture, in which a system principal, such as a user or a computer, accesses the system resources based on *a priori* assigned quota and access privileges. Hierarchical resource management is necessary for many critical reasons. Yet it imposes significant burden on users in different administrative domains trying to share their resources. Exceptions for access rules will need to be made for those users, *e.g.*, opening certain ports for certain protocols, special access permissions for certain external users from certain remote hosts, *etc.* The increasing number of exceptions would eventually create significant security hazards, or set an upper bound on the level of P2P resource sharing activities that a system can accommodate.

[4] pointed out the importance of access control and accountability in the distributed resource management system. [5] proposed an auditing scheme to enforce fair resource sharing in peer-to-peer systems using digital signatures [6] as the evidence. GridBank [7] is proposed to provide the accounting ability to GRID computing. Public-key cryptography and various electronic cash related concepts are employed to meet certain policies. The e-cash properties cited in [8], such as the *security, privacy, off-line operations, transferability, divisibility, etc.* are also very useful for distributed resource management. Although some distributed resource management systems claimed the use of electronic cash as the accounting tool for fair exchange, only few of them follow the formal e-cash framework [8] for system designs. [9][10] proposed the solution mapping from the double-spending problem in e-cash literature to the detection of the mobile agent cloning problem based on these properties. An e-cash crypto system can be used as an accounting tool in a distributed system because it is essentially the kind of signatures that have the *one-time anonymous proxy signature delegation* property [9]. That is, when a user receives a coin signed by a bank, the user can sign/spend the coin one time without revealing the

user identity in the signed coin. Any further signing of the coin will expose the identity of the user. In this paper, we adopt the generalized techniques and notations of [9], which generalizes many existing e-cash systems [11]-[17] into a single common framework.

P2P resource sharing is more flexible than its hierarchical counterpart. However, it is much more difficult to manage the credentials of the P2P nodes without any hierarchical structures. To overcome this difficulty, in this paper we propose an evidence based resource management scheme, so that once the P2P nodes receive the *resource tokens*, which are constructed by an *e-coin* scheme, from their administrators, they can directly trade for services without having to rely on other conventional resource management infrastructures during their operations. The redeeming or transfer of resource services is recorded in the *evidence chains*, which are derived from the interactions between the two nodes involved in the trading. These evidence chains can be systematically examined to detect and identify the P2P nodes which violate the resource access policies.

## 2. GEC (Generalized E-Coin)

The objective of this paper is to design a scalable and flexible management framework for accountable resource sharing in a P2P computing environment. Before we get into details of the system design, first we need to introduce some basic notations of the generalized e-coin (GEC) model [9]. Based on our construct, we can select any suitable e-coin system to realize specific functions and parameters. A GEC model is depicted in Figure 1. The identity  $U$  of a resource provider is first authenticated by the credential authority. The credential authority uses its private key  $\bar{v}$  to conduct a multi-round protocol with the provider to create a warrant  $w$  and a secret  $k$  to the provider, where the credential authority may or may not have the knowledge on  $w$  and  $k$ . The provider verifies that the resource token  $t = (w, U, k)$  satisfies the prescribed relationship  $g'(t) \rightarrow \{0,1\}$  using the public key  $v$  of the credential authority.

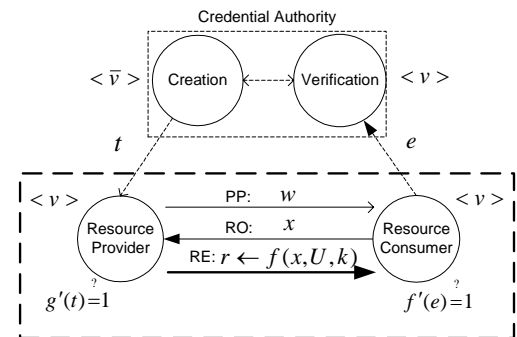


Figure 1. A GEC model without any binding.

The warrant is a system attribute signed by the credential authority that can be verified by anyone for its authenticity. The resource provider sends the warrant  $w$  to the consumer. Then the consumer randomly generates a challenge  $x$  to the provider. The provider computes the response  $r$  as a prescribed relationship  $f(x, U, k)$  and sends  $r$  to the consumer. The consumer verifies that the evidence piece  $e$  satisfies the prescribed relationship  $f'(e) \rightarrow \{0,1\}$ , where  $e = (w, x, r)$ , using the public key  $v$  of the credential authority.

The verification function  $f'(e)$  can be true only if two security criteria can be met: (1) *Legitimacy* of the resource token: only the credential authority with the correct private key  $\bar{v}$  can create a valid resource token; (2) *Ownership* of the resource token: only the peer with the correct secret  $k$  and the identity  $U$  of the warrant  $w$  can create the valid *evidence piece*  $e$ . These two security criteria imply that the provider is an authorized peer to generate the evidence. Whenever necessary, a resource consumer can submit its evidence piece to the credential authority for verification.

The construction process of resource token and resource evidence can be viewed as a *one-time anonymous proxy signature delegation* process. The credential authority is the original signer of the warrant  $w$ . The provider  $U$  is the proxy signer on behalf of the credential authority, signing  $w$  with the secret  $k$  delegated by the credential authority. The consumer is the verifier of the proxy signature (evidence)  $e$  using the public key  $v$  of the credential authority. It is infeasible to impersonate the proxy signer without the secret  $k$  of the proxy signer, to forge a legitimate delegation  $t$  without the private key  $\bar{v}$  of the credential authority, nor to forge  $e$  without the private key  $\bar{v}$  of the credential authority.

The GEC system has the one-time anonymity for its signature. That is, if the warrant  $w$  is signed once only, it is infeasible to reveal  $U$  from  $e$ . However, given two proxy signatures  $e = (w, x, r)$  and  $e' = (w, x', r')$  of the same warrant  $w$ , one can uniquely identify  $U$  from the two distinct challenge-response pairs  $(x, r)$  and  $(x', r')$ . Based on the properties of one-time anonymous proxy signature delegation, the evidence owner cannot be identified unless a double-commitment for resource access delegation occurs.

### 3. System Model

On the basis of the GEC paradigm [9], we develop a P2P resource management framework based on the system architecture depicted in Figure 2. Details of the major system operations will be presented shortly after we first define the key notations of system elements. We note that in this paper we are primarily concerned with

the exchanges of the *resource access rights* (RAR) between the peer nodes. The actual redemption of the RAR by a node to access the resources is not explicitly discussed.

For scalability, we adopt a hierarchical architecture for the credential authority nodes, which are responsible for the initial authentication and authorization of the P2P nodes to participate in the resource sharing. They are considered to be responsible for the detection of fraudulent events in this work. The resource token is used as the credential for the nodes to process the *delegation* of RAR from the *resource provider* to the *resource consumer*, i.e., the generation of the *evidence piece* and the *evidence chain*. An *evidence piece* is a one-step RAR delegation proof generated by the resource provider (with its resource token) to its consumer, which constitutes the tail of the consumer's evidence chains. An *evidence chain* is the collection of the current piece plus any previous evidence pieces starting from the head(s) (of the evidence chains) to the current node. The evidence chain carries resource management information, such as policies, credit amounts, shared resources, and evidence on the propagation path of the resource token, etc.

**Definition 1:** A resource token is a triple  $t = (w, U, k)$ , where  $w$  is the *warrant*,  $U$  is the *user identity* of the token owner, and  $k$  is the *user secret* of the token

**Definition 2:** An evidence piece is a triple  $e = (w, x, r)$ , where  $w$  is the *warrant* from a resource token,  $x$  is the *user challenge* from the consumer, and  $r$  is the *response* from the provider.

We note that there exists a subtle difference between the semantics of the conventional e-coins and the resource token/evidence. In the conventional e-coins, the consumer gives the coin to the service provider to redeem the service or the goods. On the other hand, in our system resource token is used for the evidence generation (designed from the e-coin construct) to prove that the resource provider is authorized to delegate the RAR to the consumer. As one can see from the example of  $r$ -binding depicted in Figure 9, the direction of sending resource attribute  $\Omega$  is the same as sending the evidence piece  $e$ .

After a node receives a resource token and the *credit policy* it can delegate its *resource credit* to other nodes without the participation of the third party. The credit policy sets constraints on the resource access and its delegation using the associating resource token controlled by the credential authority. The resource provider and its resource consumer directly generate the evidence piece of the delegation.

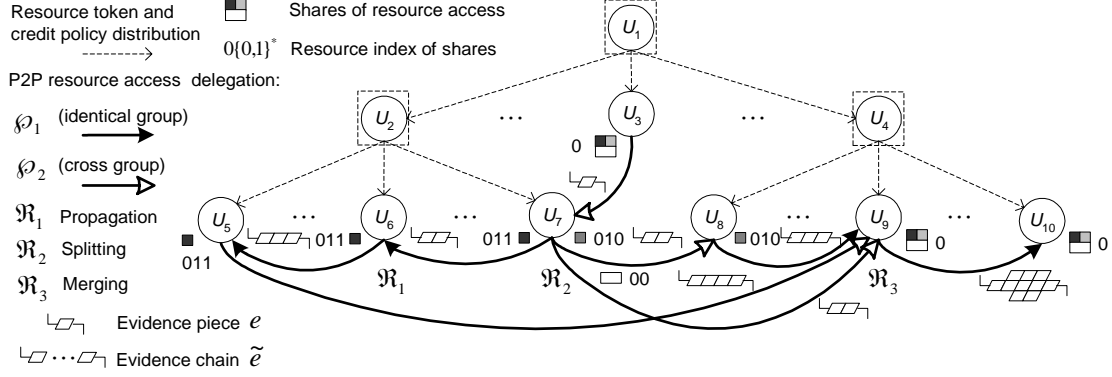


Figure 2. The system architecture of the accountable P2P resource sharing scheme.

**Definition 3:** A *delegation parameter* is a triple  $d = (\xi, \Omega, \Phi)$ , where  $\xi$  is the *credit policy* associated with the resource token,  $\Omega$  is the *resource attribute* from the provider,  $\Phi$  is the *resource order* from the consumer during the delegation process.

A resource token  $t$  associated with the credit policy  $\xi$  is legitimate if it satisfies the *token function*  $g'(t, \xi) = 1$ . An evidence piece  $e$  with delegation parameter  $d$  is legitimate if it satisfies the *evidence function*  $f'(e, d) = 1$ .

**Definition 4:** The *delegation* of the RAR from node X to node Y generates a legitimate evidence piece  $e$  from the triple  $(t, x, d)$ , where  $t$  is the resource token of X,  $x$  the user challenge from Y, and  $d$  the delegation parameter between X and Y. The new evidence piece needs to be attached to the evidence chain from X to form a new evidence chain in Y, if the delegation is related to the evidence chain mentioned above.

Once a piece of the RAR is delegated from node X to node Y, X can no longer use the delegated resource token to perform any RAR processing functions, unless the resource token is based on a type of divisible e-coin. Violation of the rule will subject X to public exposure by the fraud detection system (of the credential authority.) After the delegation is completed, Y can further delegate the RAR to another node Z, subject to the same delegation rules mentioned earlier. Node Y is called a *credit relay* and this operation is called the *propagation* of evidence.

Unless the service providers offended the single-spend rule, the evidence chain will remain anonymous, meaning that identities of the honest relays cannot be decoded from the evidence. It is also important to maintain the integrity of the evidence chain over a series of delegations using the public information of the credential authority independent of the relays.

After a certain amount of the resource credit is authorized to a node, the node should have the ability to divide the resource into smaller pieces for further operations (access, delegation, etc.) Each share of the split credit is marked by an encoded *resource index* to depict its relationship with the original evidence. This operation is called the *splitting* of resource evidence. Reversely, one must have the ability to merge the smaller pieces of e-coins back to an equivalent amount of credit carried by one single e-coin, called the *merging* of resource evidence.

The delegation is invalid if the resource provider violates the policy constraints, and the consumer may reject the evidence. If a traceable coin is used, the consumer may submit the evidence for the invalid delegation for investigation and reveal the identity of the provider if necessary.

When an evidence piece is being passed between the nodes, an *evidence chain* over a series of delegations will be formed across the nodes that have processed the evidence. Since only nodes with the correct secret knowledge of the resource token can generate the evidence piece, the records inside the evidence chain are non-deniable. When the evidence chains are collected, the credential authority can verify the fraudulent events occurred during the resource access or its delegation process.

An evidence chain  $\tilde{e}_i$  generated from a resource provider X that uses  $N$  previous evidence chains  $\tilde{e}_{i-1,1}, \dots, \tilde{e}_{i-1,N}$  to a resource consumer Y is the collection of the new evidence piece for the current delegation, plus the evidence chains of X generated from the previous delegations. The generation of the evidence chain can be defined by using an *evidence chain generator*  $E$ ,

$$\tilde{e}_i = E \left( \left( \begin{array}{ccc} \tilde{e}_{i-1,1} & e_{i-1,1} & d_{i-1,1} \\ \vdots & \vdots & \vdots \\ \tilde{e}_{i-1,N} & e_{i-1,N} & d_{i-1,N} \end{array} \right), j \right),$$

where  $e_{i-1,m}$ , for  $m = 1, \dots, N$ , are the evidence pieces that X received from the previous delegations,  $d_{i-1,m}$  are the corresponding delegation parameter, and  $j$  is a binary encoding called the *resource index*.

The specific form of the function  $E$  is determined by the nature of the e-coin system being used for implementation. While the resource token and resource evidence can prevent unauthorized access of the resource, it is difficult to prevent the duplication of the resource token for unauthorized resource access or credit delegation. We call it *double-commitment* if the shares of resource access or delegation committed by a peer exceed the total shares in its authorized possessions. In this work, we focus on the after-the-fact detection mechanism of the double-commitment events.

Now we discuss the example illustrated in Figure 2. Credential authorities  $U_1, U_2, U_4$  first distributed the resource tokens to every peer node. Then the user node  $U_3$  delegated a complete share of the RAR, which was marked by a resource index “0”, to  $U_7$ , and the delegation between them produced one single piece of evidence, *i.e.*, the evidence chain in  $U_7$  has one single evidence piece. Then  $U_7$  split (denoted by  $\mathfrak{R}_2$ ) the received RAR (from  $U_3$ ) into three pieces, indexed by “00” (1/2 share), “010” (1/4 share) and “011” (1/4 share), and delegated them to  $U_9, U_8$  and  $U_6$ . The share marked by “011” was further propagated from  $U_6$  to  $U_5$  (at  $\mathfrak{R}_1$ ),  $U_5$  to  $U_9$ . The share marked by “010” was propagated from  $U_8$  to  $U_9$ . The evidence chain size increases by one for each delegation, to include the embedded user identity of the credit.  $\mathfrak{R}_3$  denotes a merging operation in  $U_9$  where multiple evidence chains marked by “00, 010, 011,” were merged into a single evidence chain to  $U_{10}$ , which represents the delegation of the original full share of resource access, with resource index “0.” Although the evidence chains in  $U_7$  and  $U_{10}$  both represent the same RAR and resource index, the evidence chain in  $U_7$  carries the encrypted identity of  $U_3$  only, while the evidence chain in  $U_{10}$  carries identities of  $U_5$  to  $U_9$ .

#### 4. RESOURCE ACCESS EVIDENCE

Three different roles exist in an evidence based resource management system with respect to any piece of resource token and resource evidence: the credential authority, the resource provider and the resource consumer. The resource token creation and the evidence verification can be done by one single entity or multiple ones. Three basic operations among them include the distribution of resource token and credit policy, the delegation of resource access, and the collection/processing of the evidence.

For a node to work as a resource provider, it must be authenticated and authorized by the credential authority to receive a number of resource tokens and their associating credit policies to engage in the resource sharing with other nodes. The credit policy of a resource token specifies the constraints on the resource access and delegation under the credential authority. Within the policy constraints, a node has the rights to access the selected resource, and to delegate this selected resource with suitable resource token in its possessions.

It takes a three-way handshaking message exchange to execute an RAR delegation. In the first phase, the resource provider sends a *policy proposal* (PP) to the resource consumer to show its intention to delegate its resource access. In the second phase, the resource consumer acknowledges the resource provider by sending back a *resource order* (RO). In the third phase, the resource provider delegates the RAR to the resource consumer by generating the *resource evidence* (RE) using the resource token and RO. After receiving the resource evidence, the resource consumer can delegate the received RAR, or submit the evidence to the credential authority for fraud detection. The three different line styles used in Figure 3 for the three types of operations will be used in the subsequent discussion the same way as above.

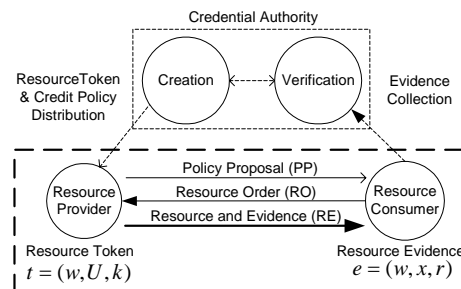
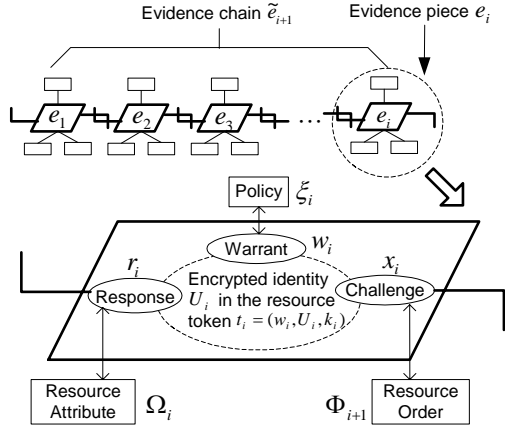


Figure 3. Basic operations of nodes.

Only the resource token owner has enough secret knowledge to generate the evidence piece with the resource token. So no node can impersonate others to create the evidence. Because of the prior authentication between the resource provider and the credential authority when the resource token is created, we can verify from the evidence that the resource consumer is an authorized user under the associating credit policy. The evidence needs to be distinguishable even after it is delegated multiple times. By using the consumer specific RO one can generate the distinct evidence pieces even when the same RAR is delegated several times.

On the basis of the GEC paradigm, which generalizes many existing e-cash systems into a single common framework, we develop the resource token and resource evidence. The basic construct of the evidence piece and the evidence chain is depicted in Figure 4.



- (1) Encrypt the identity  $U_i$  from the resource token  $t_i = (w_i, U_i, k_i)$  to the evidence piece  $e_i = (w_i, x_i, r_i)$ .
- (2) Bind the delegation parameter  $d_i = (\xi_i, \Omega_i, \Phi_{i+1})$  to the evidence piece  $e_i = (w_i, x_i, r_i)$ .

Figure 4. Evidence piece and evidence chain construction through e-coin binding.

The resource token is composed of a warrant, a user identity and a secret from the credential authority. Through the zero-knowledge proof [21] of the e-coin system, the resource token is transformed to an evidence piece composed of a warrant (from the credential authority), a challenge (from the consumer) and a response (from the provider).

We can verify from the evidence piece that an authorized but encrypted user identity is embedded into this warrant-challenge-response component using the public information of the credential authority.

Each evidence piece represents an unforgeable relationship between the encrypted identity  $U$  of the resource provider, the credit policy  $\xi$  associated with the resource token, the resource attribute  $\Omega$  from the provider and the resource order  $\Phi$  from the consumer, in a single delegation process. The cryptographic linkage is important to ensure the integrity of the delegation history inside the evidence chain.

Referring to Figure 5, which depicts an evidence chain propagation scenario, we denote an evidence chain generated from  $U_A$  to  $U_B$  as  $\tilde{e}_B \leftarrow E((\tilde{e}_A, e_{A \rightarrow B}, d_{A \rightarrow B}))$  where  $E(\cdot)$  is a one-way evidence chain generator,  $\tilde{e}_A$  is the last evidence chain that the provider  $U_A$  received from the last delegation,  $e_{A \rightarrow B}$  is the evidence piece generated between A and B,  $d_{A \rightarrow B}$  is the corresponding delegation parameter of  $e_{A \rightarrow B}$ , and  $j$  is the resource index. If  $U_A$  is the first provider in the chain, then  $\tilde{e}_A \leftarrow \phi$ .

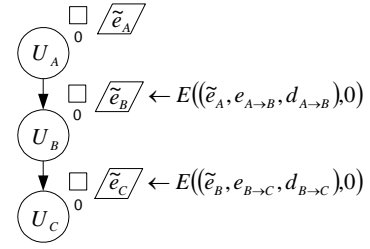


Figure 5. Evidence chain generation under the resource evidence propagation operation.

The resource index  $j$  is a binary encoding of the shares of the resource access. A complete share of the resource access is denoted by “0.” Each subsequent bit represents subdivision of the share by half. The resource index is useful in splitting operation to distinguish the size and location of the shares. Figure 6 depicts the splitting of the evidence chain. The graph at the left hand side of Figure 6 depicts the splitting of a resource block into multiple sub-blocks, where each bit represents a “divide by 2” operation, and the graphs at the right hand side depict the corresponding evidence chains.

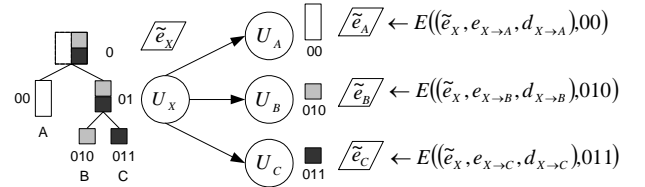


Figure 6. Evidence chain generation under the resource evidence splitting operation.

For  $N$  evidence chains to combine into a single evidence chain in the merging operation, the first parameter is represented by an  $N$ -row matrix, as shown in Figure 7.

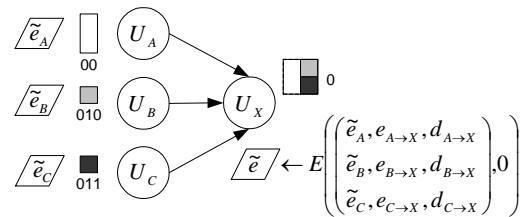


Figure 7. Evidence chain generation under the resource merging operation.

## 5. EVIDENCE BASED P2P RESOURCE ACCESS

In this section, we show how to construct the P2P resource access evidence based on the GEC model. Later, we will show how to select suitable existing e-coin systems, such as [12][13] to implement our scheme. Essentially an e-coin is a binary string with a special signature of a certain denomination. Existing e-coin schemes only provide the mechanisms to verify the

authenticity of the encrypted identities. The events that different entities contribute to the resource attributes, credit policy and the resource order contributed in the delegation process cannot be reflected. To overcome these problems, we propose three attribute binding techniques of GEC for different resource delegation scenarios, so that the credit policy (distributed by the credential authority), the resource attribute (sent from the provider) and the RO (acknowledged from the consumer) can be cryptographically tied with the warrant, the response and the challenge respectively in the e-coin for the generation of the anonymous evidence on the delegation events.

### 5.1 $w$ -Binding

The basic idea of  $w$ -binding, see Figure 8, is to modify  $g'(t)$  to  $g'(t, \xi)$  in the GEC model, where  $\xi$  is the credit policy enforced by the credential authority. For this change, it is necessary to modify the verification function  $f'(e)$  to  $f'(e, \xi)$ . Although the secret  $k$  may also be modified, we call it  $w$ -binding because only  $w$  is explicitly transferred from the provider to the consumer.

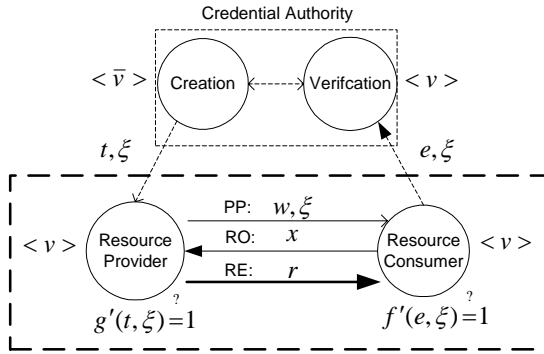


Figure 8.  $w$ -binding of the credit policy  $\xi$  from the credential authority.

The  $w$ -binding is a static binding technique used by the credential authority to grant the resource provider a credit policy embedded into the resource token. It also represents the indirect control of the credential authority that restricts the provider's behavior on the resource access and its delegation. We categorize  $w$ -binding as a static technique because the binding item is fixed at the resource token creation time.

We can generally take either an external or internal construction approach for the transformation according to the GEC model, among other variations for the particular e-coin systems. In the external construct approach, we first follow the conventional way to construct the warrant  $w$ . Then the credential authority additionally gives a signature  $sign(w, \xi')$ , where  $\xi'$  is a credit policy. The new credit policy in the  $w$ -binding is essentially a signature pair  $\xi = (\xi', sign(w, \xi'))$  [11]. For the internal construct

approach, the credit policy  $\xi$  is directly incorporated into the warrant  $w$  or the secret  $k$ .

Certain public parameters or hash values are required to attach the credit policy  $\xi$  to the resource token. The two potential solution approaches, *public redundancy* and *hash redundancy* have been commonly used for providing the one-way feature for certain secure constructs, such as prevention of forging the token. The public redundancy approach refers to using of a public parameter, such as the public generator of the discrete log. Suppose  $\hat{\xi}$  is a suitable public parameter used in both  $f'$  and  $g'$ . Our goal is to use a suitable one-way mapping  $g$  to transform the credit policy  $\xi$  to the domain of  $\hat{\xi}$ , where  $\hat{\xi}$  is a piece of derived knowledge from  $\xi$ . The policy holder can compute  $\hat{\xi}$  from the transfer policy  $\xi$  during the RAR delegation process.

The basic idea of hash redundancy is to embed additional information of the credit policy into the hash value without compromising the original properties. Suppose  $\hat{\xi}$  is some parameter to construct  $(w, k)$ , while  $hash(\hat{\xi})$  is the hash value used in both prescribed relationships  $f'$  and  $g'$ , we would replace  $hash(\hat{\xi})$  by  $hash(g(\hat{\xi}, \xi))$  where  $g$  is a suitable one-way function without increasing the collision probability of  $hash(\hat{\xi})$ . This way we make  $\xi$  an integrated element in the processes of resource token construction and evidence verification.

Under normal conditions, the identity of an honest resource provider can be concealed in the evidence. We note that certain level of traceability of the resource evidence may be desirable for resource management. If a credit policy is shared by a group, then the credential authority can not trace individuals within the group. On the other hand, if full untraceability of resource is needed, one can use blind signature in the external construction approach to achieve this goal. In this case, the credential authority is knowledge-free from the credit policy. We will show later how this zero-knowledge  $w$ -binding can combine with  $r$ -binding to manage the evidence splitting operations.

### 5.2 $r$ -Binding

The resource token helps the provider to generate the resource evidence to prove that it is an authentic user, authorized by the credential authority to delegate the RAR, while keeping the user incognito.  $r$ -binding is considered a dynamic binding scheme, because it can be performed by a resource provider to select the resource attribute to be bound with an evidence piece after it has been issued by the credential authority.

The  $r$ -binding protocol is depicted in Figure 9. Although  $r$ -binding allows a resource provider to select the resource attributes for the binding, its choices are still subject to other constraints defined in  $w$ -binding and  $x$ -binding, making it possible to define the global and the local resource policies within one single resource evidence.

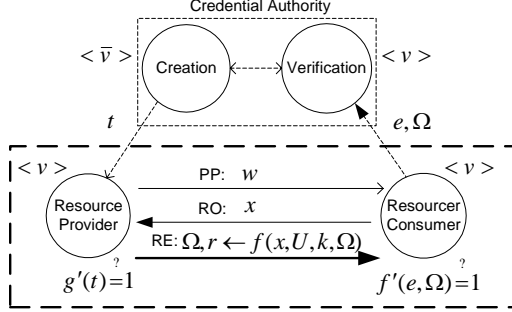


Figure 9.  $r$ -binding of the resource attribute  $\Omega$  from the resource provider.

As the RAR is delegated, its generated evidence represents the commitment of the resource provider to give the RAR to the resource consumer. One cannot impersonate the commitment to without knowing the secret of the resource token. The  $r$ -binding is unforgeable without the secret knowledge of the credential authority. Since the credential authority does not participate in both the RAR delegation and  $r$ -binding processes, it is suitable for decentralized resource management.

The basic idea of  $r$ -binding is to replace  $r = f(x, U, k)$  in GEC by  $r = f(x, U, k, \Omega)$ , and hence the name  $r$ -binding. Accordingly, the verification function  $f'(e)$  needs to be changed to  $f'(e, \Omega)$ , see Figure 9. In some cases, a derivative of  $\Omega$  may be used in the binding and the verification processes.

One possible approach for  $r$ -binding is to replace the random challenge  $x$  in the GEC model by  $hash(x, \Omega)$ . This is a plausible approach because  $x$  can be any random string by its nature. We replace  $x$  with  $hash(x, \Omega)$  which is also random in nature, but cryptographically tied with  $\Omega$ . The response function becomes  $r = f(hash(x, \Omega), U, k)$  and the verification becomes  $f'(w, hash(x, \Omega), r)$ . Other ways are possible in specific schemes such as [12][13].

### 5.3 $x$ -Binding

Similar to the operations of  $w$ -binding, which is controlled by the credential authority, and  $r$ -binding, which is controlled by the resource provider, the  $x$ -binding is a resource attribute binding technique controlled by the resource consumer. It is considered a dynamic binding technique because such a binding takes

place after the resource token has been issued by the credential authority. Referring to Figure 10, instead of randomly generating the challenge  $x$ , the resource consumer generates the challenge  $x_\Phi = hash(x, \Phi)$  to incorporate the RO,  $\Phi$ , into the “challenge” based on the GEC model, hence the name of  $x$ -binding. We use  $e_\Phi = (w, x_\Phi, r)$  to denote the evidence piece in this scheme.

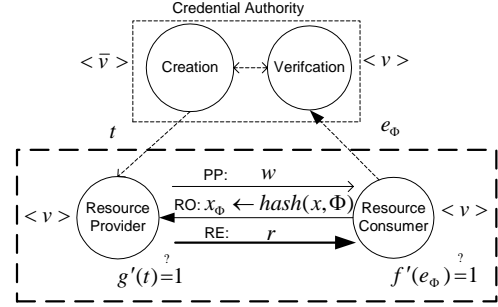


Figure 10.  $x$ -binding of the resource order  $\Phi$  from the resource consumer.

Similar to  $r$ -binding,  $x$ -binding allows the resource consumer to choose the entity to be bound with  $x$ . However, unlike the  $w$ -binding and  $r$ -binding,  $\Phi$  is not necessary to be explicitly transferred through the network because the resource consumer has enough information for the verification. In many cases,  $x$ -binding does not have the instant implication on the current delegation when it works alone. But it may control the subsequent delegation when a series of  $x$ -binding and  $r$ -binding are used in an evidence chain. We will further elaborate the application of  $x$ -binding for multiple delegations later.

## 6. DOUBLE-COMMITMENT DETECTION

In this section, we discuss the generation process of evidence chain under propagation, splitting and merging of resource credits, without the on-line involvement of the credential authority, *i.e.*, the P2P resource sharing. Our goal is to (1) ensure the integrity of the evidence chain, (2) maintain the privacy of the lawful credit operations in the evidence chain, and (3) identify the culprit of double-commitment from the evidence chain after the fact. Our techniques are derived from the generalized transferable e-coin [9] and the generalized divisible e-coin [11] respectively. A similar technique is used for the merging operation using a self-delegation. The “zero value coin” in [9] is equivalent to the resource token. We show that the above techniques in [9][11] implicitly employ the  $w$ -binding and the  $x$ -binding to restrict the behavior of the  $r$ -binding in the three operations.

### 6.1 Evidence Chain Propagation

For evidence chain propagation, we assume that a resource provider has gathered enough resource tokens

from the credential authority. Our goal is to uniquely identify double-commitment offenders while a series of  $x$ - and  $r$ -bindings occurred during the evidence chain delegation that is originated from one particular resource token. This is primarily achieved by the fact that the double-commitment along the evidence chain will leave non-forgeable, distinct challenge-response pairs of the same warrant in the evidence pieces. The identity of the resource token owner can be identified by using the double-spending detection mechanism of the employed e-coin systems.

Figure 11 depicts the details of the evidence chain propagation operations. For the example shown in Figure 11,  $U_{i-1}$  is the  $(i-1)$ -th peer delegated to access the resource, and it is to pass the rights to  $U_i$ . The resource token  $t_{i-1} = (w_{i-1}, U_{i-1}, k_{i-1})$  is used by  $U_{i-1}$  to receive the evidence chain  $\tilde{e}_{i-1}$  to prove its authorized access from the last delegation.

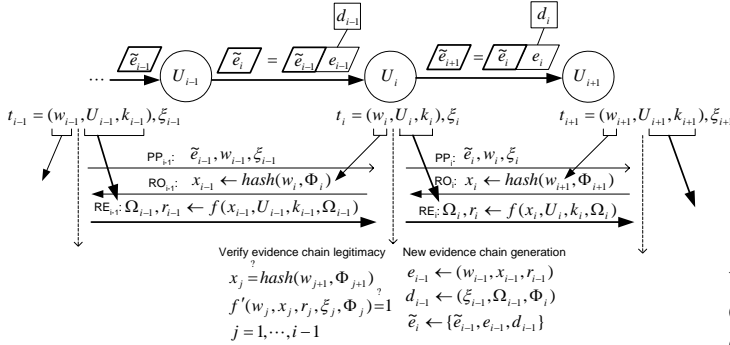


Figure 11. Evidence chain generation from  $\tilde{e}_{i-1}$ ,  $\tilde{e}_i$  to  $\tilde{e}_{i+1}$  under the propagation operation.

To delegate the RAR from  $U_{i-1}$  to  $U_i$ ,  $U_{i-1}$  sends  $\tilde{e}_{i-1}$  to  $U_i$ .  $U_{i-1}$  also sends (the warrant)  $w_{i-1}$  and credit policy  $\xi_{i-1}$  of the resource token  $t_{i-1}$  to  $U_i$ . Then  $U_i$  picks one of its unused resource tokens  $t_i = (w_i, U_i, k_i)$  and generates the challenge  $x_{i-1} = \text{hash}(w_i, \Phi_i)$  where  $\Phi_i$  is the resource order from  $U_i$ .  $U_{i-1}$  responds by  $r_{i-1}$  using the secret  $k_{i-1}$  and the challenge  $x_{i-1}$ . Then  $U_i$  verifies that the linkage between each evidence piece satisfies  $x_j = \text{hash}(w_{j+1}, \Phi_{j+1})$ , and the legitimacy and ownership of every evidence piece satisfies  $f'(w_j, x_j, r_j, \xi_j, \Omega_j) = 1$ , where  $1 \leq j \leq i-1$ . Finally, the new evidence chain is generated in  $U_i$ :  $\tilde{e}_i \leftarrow \{\tilde{e}_{i-1}, e_{i-1} = (w_{i-1}, x_{i-1}, r_{i-1}), d_{i-1} = (\xi_{i-1}, \Omega_{i-1}, \Phi_i)\}$ .

The above constructs guarantee that a node uses the same resource token to receive the old evidence chain and to generate the new one, which is a crucial security requirement for the culprit identification of the double-commitment in the evidence chains.

In the above discussion we assume that the resource token of  $U_i$  is unused, based on the generalized transferable e-coin [9], to prevent the possible infringement of credit relay privacy. Reusable resource tokens for specific e-coin schemes [12][13] can be employed, without credit relay identity being compromised.

If  $U_i$  commits the resource access delegation twice or more, it will generate two evidence chains  $\tilde{e}_{i+1}$  and  $\tilde{e}'_{i+1}$ , from which two evidence pieces  $e_i = (w_i, x_i, r_i)$  and  $e'_i = (w_i, x'_i, r'_i)$  with the same warrant but different challenge-response pairs will exist. We can uniquely identify  $U_i$  from these challenge-response pairs, as shown in Figure 12.

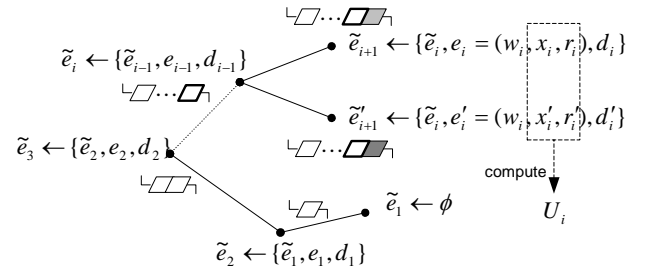


Figure 12. Double-commitment investigation under the propagation operation.

Similar to other transferable e-coin systems, the size of the resource evidence grows with the length of the evidence chain [18]. This problem can be remedied by an evidence renewal protocol, where the body of the evidence chain can be substituted by a signature on the head and the tail of the chain. Details of this technique will not be discussed in this paper due to space limit.

## 6.2 Splitting operation

The central idea in tracking the splitting of the resource credit is to tie the root of a binary credit tree with the blindly signed resource token in the external construct of the  $w$ -binding. Each tree node, whose identity is denoted by a secret value, represents certain shares of the resource access to be delegated. When the resource provider delegates the shares of access represented by the tree node, the secret values of its ancestors are exposed. The  $w$ -binding restricts the provider to select the tree nodes (the shares) using  $r$ -binding, subject to certain double-commitment rules. If the provider violates the double-commitment rules, we can compute the user identity of the provider from the exposed secrets and the challenge-response pairs from the evidence chains.

A binary credit tree is constructed for  $t$  levels, in which each node has two child nodes, where the unique root node exists at the top of the tree. Each node of the tree is represented by a secret value, called  $k$ -value. The  $k$ -value of the root is denoted by  $k_0$ . The left child and the right

child of a node, which is denoted by  $k_j$ , is represented by  $k_{j_0}$  and  $k_{j_1}$  respectively, where the *resource index*  $j$  is a binary string. Given two suitable one-way functions  $H$  and  $F'$ , the  $k$ -values are assigned as follows:

Leaf node assignment:

$k_j = H(\varepsilon \parallel j)$ , where  $\varepsilon$  is the random seed of this binary credit tree, and “ $\parallel$ ” stands for the concatenation of two entities.

Non-leaf node assignment:

$$k_j = H(H(F'(w, k_{j_0})) \parallel H(F'(w, k_{j_1})))$$

In the subsequent discussions, the root of the tree is referred to as  $K_0 = F'(w, k_0)$ , unless explicitly specified otherwise. Figure 13 demonstrates an example of the binary credit tree construction.

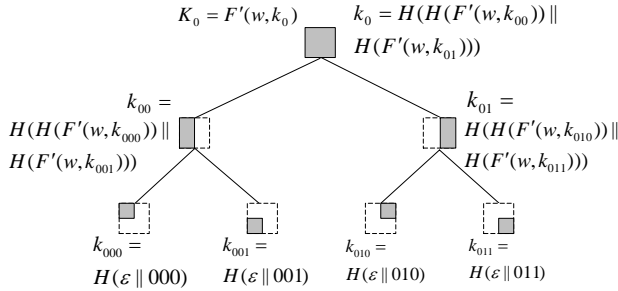


Figure 13. Binary credit tree construction for the splitting operation.

$K_0$  is tied with the warrant  $w$  of the resource token through blindly signed, external  $w$ -binding. The resource provider first receives its resource token  $t$  as usual. Then it blinds the hash value of  $w$  and  $K_0$ , and sends it to the credential authority, which then gives the blind signature to the received hash value. The resource provider then unblinds the signature as usual.

Suppose the resource being accessed can be subdivided into  $M$  shares. Then a node at the  $i$ -th level represents the portion of  $M/2^{i-1}$  shares. The resolution in differentiating details of resource sharing is determined by the  $t$  value. To avoid the shares being over-split into negligible units, we can define the policy by  $w$ -binding to enforce the minimum unit of share. To delegate the share of resource access corresponding to the tree node  $k_j$ , and the warrant  $w$ , the resource provider reveals the *need-to-know information* to the consumer for computing the tree root  $K_0$ . The need-to-know information includes:

F-value:  $F'(w, k_j)$

H-value:  $H(F'(w, \bar{k}_j))$  if the node  $\bar{k}_j$  is a direct offspring of an ancestor of the node  $k_j$ , but not on the route from the node  $k_j$  to the root node.

The need-to-know information actually reveals the  $k$ -values of all ancestors of the node  $k_j$ . Figure 14 illustrates an example to give the need-to-know information corresponding to the node  $k_{000}$ .

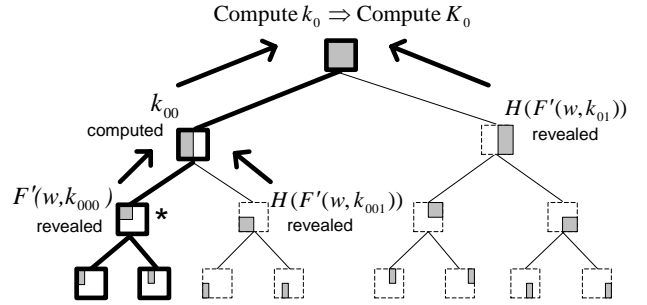


Figure 14. F-value and H-value revealed for delegation under the splitting operation.

For the splitting operation, the evidence piece is generated as follows: the resource provider first sends the warrant  $w$ , the corresponding credit policy  $\xi$  and tree root  $K_0$  and their signature  $sign(hash(w, K_0))$  to the resource consumer. The consumer verifies that  $sign(hash(w, K_0))$  is a valid signature from the credential authority and generates to the provider a challenge  $x$  using the resource order  $\Phi$ . The provider incorporates the  $k$ -value into the response  $r$  as in the generalized model using the  $r$ -binding technique. This  $k$ -value is corresponding to the shares of access that the provider intends to delegate. The provider sends the resource attribute  $\Omega$ , the response  $r$  and the resource index  $j$  of the corresponding shares to the consumer. The consumer verifies that the response is valid with respect to  $w$  and  $r$ . The consumer also verifies the correct amount of shares by checking the number of level to re-construct the root  $K_0$  from the need-to-know information.

The response  $r$  mentioned above should be constructed in such a way that, (1) given the  $k$ -value, the identity  $U$  of the provider can be computed from the single corresponding challenge-response pair  $(x, r)$ , and (2) given  $(x, r)$  and  $(x', r')$  of the same warrant  $w$  and  $k$ -values, the identity  $U$  can be computed. One possible way for such a construction is to replace the role of the secret  $k$  in the generalized model with some particular types of  $k$ -values. However, such a replacement must be handled with great care not to affect the properties of  $w$ ,

as the secret  $k$  is one of the elements to construct  $w$  during the resource token creation process.

The resource transfer is doubly committed if the traverse path of transfers violates the *route node* or *same node rule* [8][11][17][19]. The route node rule states that if a node is used for delegation, then all ancestors and all descendents of this node cannot be used for delegation anymore. The same node rule states that no node can be used for transfer more than once.

As shown in Figure 15 (a), if two nodes on the same route are used, then one node should be an ancestor of another. Let a node  $k_j$  denote an ancestor of node  $k_i$ . When  $k_i$  is used for delegation, its need-to-know information should reveal the value of  $k_j$ . When node  $k_j$  is used for transfer, we have a single challenge-response pair  $(x_j, r_j)$ . With  $(x_j, r_j)$  and  $k_j$ , we can compute the identity of the user  $U$  who offends the double-commitment rule.

Figure 15 (b) illustrates the case when a node transfers the credit more than once. Different from the route node rule, no extra  $k$ -value is revealed for the identity computation. However, we know that the two different challenge-response pairs  $(x_i, r_i)$  and  $(x_j, r_j)$  are computed from the same  $k$ -value as they have the same  $F$ -value. They can be used for extracting the identity  $U$  of the user for double-commitment.

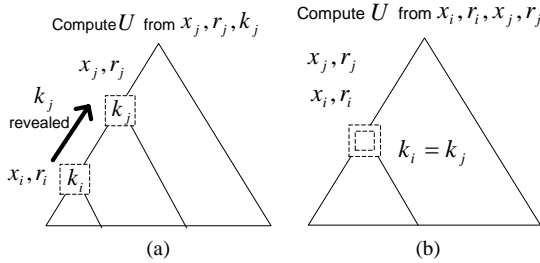


Figure 15. Double-commitment rules under the splitting operation: (a) route node rule; (b) same node rule.

The generalized resource evidence transfer and division can work in parallel in the resource access delegation. The binary credit tree structure can be used for constructing the resource token. A share split from a provider can be further split by the consumer in the subsequent delegation, provided that all the shares add up to the original sum. Again, the key step is to use  $x$ -binding to ensure that the peer uses the same resource token to receive and to delegate the rights of resource access. The  $w$ -binding and  $r$ -binding allow one to choose multiple secrets to re-delegate the resource access under the double-commitment rules without the user privacy being compromised.

### 6.3 Merging operation

The purpose of merging is to merge multiple evidence chains into a single one, so that in the future credit transfers that one can use a single evidence piece to represent the merged evidence chains. We note that this operation does not compress the size of the existing evidence chains. Evidence chain merging is done by creating a *self* transfer of the resource evidence chains to the node that is merging them, where a single resource token of the node is used for generating the same challenge for the evidence chains being merged. After the transfer, the node uses the newly generated evidence for the subsequent delegation. Figure 16 (a) and (b) compare the size of the evidence chains with and without merging operations.

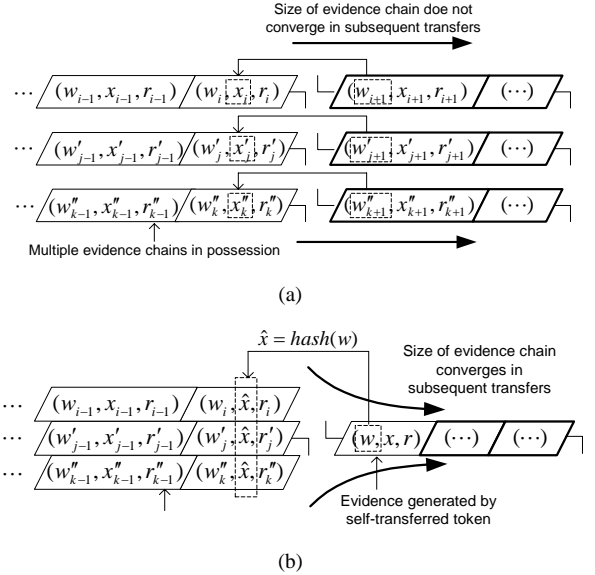


Figure 16. (a) Evidence chains without merging operation, and (b) evidence chains after merging operation.

Again, the key step is to use  $x$ -binding to enforce the user to use the same resource token to receive and to send the merged evidence. The double-commitment of the merged evidence implies the double-commitment of this self-transferred resource token. Moreover, the merged evidence can be split again if the resource token is a divisible one.

## 7. EXAMPLE: A FERGUSON E-COIN BASED IMPLEMENTATION

In this section, we demonstrate how to implement our scheme based on Ferguson's single-term off-line coin [12][13]. Similar implementations can be developed for the variations of Ferguson [10][11][16]. The Ferguson scheme assumes that all computations are done modulo  $n$  (the RSA public modulus) while those involving exponents are done modulo  $v$  unless otherwise specified.

By directly mapping the Ferguson's coin into the GEC model, and using its original notations for this section, our resource token is essentially a 7-tuple

$t = (a, b, c, U, k, S, T)$ , where  $a, b, c$  are the *warrant numbers*,  $U$  is the *user identity*,  $k$  is a *secret number*, and  $S, T$  are the *secret signatures* from the credential authority. The resource evidence piece is essentially a 6-tuple  $e = (a, b, c, x, r, R)$ , where  $x$  is a *challenge* from the resource consumer, while  $r, R$  are the *responses* from the resource provider.

The token function  $g' \rightarrow \{0,1\}$  is to test whether or not  $T^v = C^U B$ ,  $S^v = C^k A$ ,  $A = ag_1^{f_1(a)}$ ,  $B = bg_2^{f_2(h_b)}$ ,  $C = cg_3^{f_3(h_c)}$ , where  $(v, 1/v)$  is the RSA public-private key pair of the credential authority,  $g_1, g_2, g_3, h_b, h_c$  are some public parameters, and  $f_1$  is a suitable one-way mapping that can be public to all the nodes. The evidence function  $f' \rightarrow 1$  is to test whether or not  $R^v = C^r A^x B$ . The delegation function  $f$  is to set  $r = xk + U$  and  $R = S^x T$ .

Double-commitment detection is based on the polynomial secret sharing scheme [20], where the two different points on the secret sharing line  $r = kx + U$  can reveal the offender's identity  $U$ . The  $x$ - and  $r$ -bindings of Ferguson's coin are trivial. We can include the resource attribute  $\Omega$  and the resource order  $\Phi$  by replacing the response  $r = xk + U$ ,  $R = S^x T$  in the delegation function by  $r = \text{hash}(x, \Omega, \Phi)k + U$ ,  $R = S^{\text{hash}(x, \Omega, \Phi)} T$ .

The internal  $w$ -binding is demonstrated in [10], from which the public parameter  $h_b$  is replaced by  $h_b^{\text{hash}(\xi)} \bmod p$ , where  $\xi$  is the credit policy, and  $p$  is a large prime with  $p-1$  being a multiple of  $n$ . By this construct, the parameter  $h_b$  can be derived by  $\xi$ .

An evidence chain is in the form of  $\tilde{e}_i = \{\tilde{e}_{i-1}, e_{i-1}, d_{i-1}\}$  where  $e_{i-1} = (a_{i-1}, b_{i-1}, c_{i-1}, x_{i-1}, r_{i-1}, R_{i-1})$  and  $d_{i-1} = (\xi_{i-1}, \Omega_{i-1}, \Phi_{i-1})$ . The transferable solution in [10][16] links the evidence pieces together to form the evidence chain in the propagation operation. We define the linkage of the evidence pieces in the evidence chain can by setting the challenge  $x_j = \text{hash}(a_{j+1}, b_{j+1}, c_{j+1}, \Phi_{j+1})$  for each  $j$  in the evidence chain. If there is a double-commitment, we can collect the two evidence chains, with the evidence pieces of the same warrant  $w_j = (a_j, b_j, c_j)$  but different challenge-response pairs  $(x_j, r_j)$  and  $(x'_j, r'_j)$ , which immediately identify the offender  $U_j$  by interpolation from the secret sharing line.

The divisible token for splitting operation using Ferguson's coin is shown in [11]. The  $F$ -value for binary credit tree construction is set to be  $K_j = A^{k_j}$  where  $k_j$  is the  $k$ -value of the node with the resource index  $j$ . [11] assigns the secret number  $k = 1$  and replaces its role by the  $k$ -value of the delegation node. To tie the root  $K_0$  with the

resource token  $t = (a, b, c, U, k, S, T)$ , an external  $w$ -binding is perform to yield a blind signature  $\text{hash}(K_0, A, B, C)^{1/v}$  from the credential authority. To delegate the share of resource access with resource the index  $j$ , the delegation function is modified to  $r = Ux + k_j$ ,  $R = T^x S^{k_j}$ . The evidence function is modified to  $R^v = C^r B^x K_j$ . On the double-commitment, if a peer violates the route node rule with nodes  $k_i$  and  $k_j$ , then the  $k$ -value of the ancestor node  $k_j$  is exposed when node  $k_i$  is used for the delegation. An offender's identity  $U$  can be computed from  $r_j = Ux_j + k_j$  given the challenge-response pair  $(x_j, r_j)$  in the evidence chain. If the peer violates the same node rule, where the node  $k_j$  is used twice, then the two challenge-response pairs  $(x_j, r_j)$  and  $(x'_j, r'_j)$  from the evidence chains can be used to reveal  $U$  by the interpolation of  $r_j = Ux_j + k_j$  and  $r'_j = Ux'_j + k_j$ . The technique for merging operation in Ferguson's coin is similar to the one in the propagation. We compute  $x_j = \text{hash}(a_{j+1}, b_{j+1}, c_{j+1}, \Phi_{j+1})$  as the common challenge to generate a single evidence chain from multiple evidence chains in possessions.

## 8. CONCLUSION

Evidence based resource access is suitable for a P2P computing environment, in which the computing nodes can use the resource credentials issued by the credential authority to exchange and share the computing resources with minimal intervention from the credential authority. We note that while our solutions are built upon the generalized e-coin paradigm, careful analysis of the semantics of each and every operation and system variable is a non-trivial undertaking. Proper applications of the three operational primitives (propagation, splitting merging) of the resource evidences can support complex resource access delegation scenarios. We demonstrate the feasibility of our scheme on the Ferguson's e-coin system to construct an evidence based P2P resource management scheme. Similar constructs can be obtained for other e-coin schemes.

This paper represents our first step to investigate this interesting and important area of P2P resource management. Much more can be and should be done in the future. For instance, the level of security and privacy depends on the specifications of the particular e-coin schemes and the resource management applications. Further improvement of the evidence chain structures to reduce the overhead is a must. Detection of fraudulent incidents with minimal overhead is critical to system integrity. These, and many other related issues, are the issues to be investigated in the near future.

## REFERENCES

- [1] N. Minar, *Distributed Systems Topologies*, Openp2p.com, 2001  
[http://www.openp2p.com/pub/a/p2p/2001/12/14/topologies\\_one.html](http://www.openp2p.com/pub/a/p2p/2001/12/14/topologies_one.html)
- [2] P. Mockapetris. *Domain Names – Implementation and specification*. Internet RFC 1035.
- [3] Foster and C. Kesselman, editors., *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, San Francisco, 1998.
- [4] E. Adar and B. Huberman. *Free riding on Gnutella*. Firt Monday, 5(10), Oct, 2000.
- [5] T. Ngan, D. Wallach and P. Drushel, “Enforcing Fair Sharing of Peer-to-Peer Resources.” *2nd International Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.
- [6] J. FeigHHI and P. Williams, *Digital Certificates: Applied Internet Security*. Addison Wesley Longman Inc., 1998. ISBN 0-201-30980-7, 1998.
- [7] A. Barmouta and R. Buyya, “GridBank: A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration.” *17th Annual International Parallel & Distributed Processing Symposium (IPDPS 2003) Workshop on Internet Computing and E-Commerce*, 2003.
- [8] T. Okamoto and K. Ohta., “Universal Electronic Cash.” *In Advances in Cryptology – CRYPTO’91*, pp. 725-729, 1991.
- [9] T. C. Lam and V. K. Wei, “Mobile Agent Clone Detection using General Transferable E-Cash.” *International Conference on Information Security (InfoSecu’2002)*, 2002.
- [10] T. C. Lam and V. K. Wei, “A Mobile Agent Clone Detection System with Itinerary Privacy.” *In Proceedings of the Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE ’02)*, pp. 68 - 73, 2002.
- [11] T. Eng and T. Okamoto, “Single-Term Divisible Electronic Coins.” *In Advances in Cryptography – Proceedings of EUROCRYPT ’94*, LNCS 950, pp. 306 – 319, Springer-Verlag, 1995.
- [12] N. Ferguson, “Single term off-line coins.” *In Advances in Cryptology – Proceedings of EUROCRYPT ’93*, LNCS 765, pp. 318 – 328, Springer Verlag, 1993.
- [13] N. Ferguson. Single term off-line coins. Technical Report CS-R9318, CWI (Centre for Mathematics and Computer Science), Amsterdam, 1993.
- [14] N. Ferguson, “Extensions of single-term coins.” *In Advances in Cryptology – CRYPTO ’93*, LNCS 773, pp 292 – 301, Springer Verlag, 1993.
- [15] S. Brands, “Untraceable Off-line Cash in Wallet with Observers.” *In Advances in Cryptology – CRYPTO ’93*, pp. 302 – 318, 1993.
- [16] H. Y. Wong, *Issues in Electronic Payment Systems: A New Off-line Transferable E-coin Scheme and a New Off-line E-check Scheme*. Master’s Thesis, Department of Information Engineering, The Chinese University of Hong Kong, 2001.
- [17] Y. Tsiounis, Y. Frankel, and A. Chan, “Easy come – easy go divisible cash (updated version, GTE Tech report).” *EUROCRYPT ’98*, LNCS, pp. 561-575, 1998.
- [18] D. Chaum and T. P. Pedersen, “Transferred Cash Grows in Size.” *EUROCRYPT ’92*, pp. 390 -407, 1992.
- [19] T. Okamoto, “An Efficient Divisible Electronic Cash Scheme.” *In Advances in Cryptology – Proceedings of CRYPTO ’95*, LNCS 950, pp. 438 – 451, Springer Verlag, 1995.
- [20] A. Shamir, “How to share a secret.” *Communications of the ACM*, 22(11) pp. 612-613, 1979.
- [21] A. J. Menezes, P. C. V. Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, Fla, 1997.