THEMIS†: Towards Mutually Verifiable Billing Transactions in the Cloud Computing Environment

Ki-Woong Park†, Sung Kyu Park†, Jaesun Han‡, and Kyu Ho Park†
Korea Advanced Institute of Science and Technology†, NexR Co.Ltd‡
{woongbak, skpark, kpark}@core.kaist.ac.kr, and jshan@nexr.co.kr

Abstract

The ability to record and keep account of the usage of cloud resources in a credible and verifiable way is a precursor to widespread cloud deployment and availability because usage information is potentially sensitive and must be verifiably accurate. In an attempt to provide a mutually verifiable resource usage and billing mechanism, we found that the frequent asymmetric key operations of a digital signature lead to excessive computations and a bottleneck of billing transactions. As a remedy for these limitations, we propose a mutually verifiable billing system called THEMIS. The system, which introduces the concept of a cloud notary authority for the supervision of billing, makes billing more objective and acceptable to users and cloud service providers. THEMIS generates mutually verifiable binding information that can be used to resolve future disputes between a user and a cloud service provider. Because THEMIS does not require any asymmetric key operations of users and providers, it provides a level of security that is identical to that of a Public Key Infrastructure (PKI) and it minimizes the latency of billing transactions. This work has been undertaken on a real cloud computing service called iCube Cloud [1].

1. Introduction

The pay-as-you-go pricing model is one of the core components of the cloud computing environment. It enables users to scale their capacity upwards or downwards as their computing requirements change. Cloud services change the economics of computing by enabling users to pay only for the capacity that they actually use. In this environment, the ability to record and account for the usage of cloud resources in a credible and verifiable way is a precursor to widespread cloud deployment and availability [1, 23]. On the basis of verifiability, cloud service providers (CSPs) and users can both construct credible, verifiable usage records to prove which resources were allocated and when they were initiated. On the other hand, in commercial cloud services, such as Amazon EC2 [14] and Azure [3], the billing transactions and management are both processed by the CSP alone; there is no mutual verifiability of billing transactions.

A PKI-based digital signature [17] stands out as a fundamental and widely used mechanism for enforcing the verifiability of billing systems. However, the computational complexity of a PKI may result in a high computational overhead and intolerable billing response time because asymmetric key operations for digital signatures need to be performed with regard to both the thin client terminal and the CSP. In addition, the amount of billing transactions increases rapidly in proportion to the number of users that require diverse cloud resources. By thoroughly investigating conventional cloud billing systems, we identified the following two fundamental requirements, which drive the architecture of our billing system:

- **A fine-grained billing mechanism with a minuscule computational overhead:** The frequent transactions of fine-grained, mutually verifiable billing lead to an excessive computational overhead or billing system bottleneck. To mitigate these problems, we propose a computationally efficient billing scheme called THEMIS, which provides mutual verifiability. The proposed billing scheme extremely minimizes the asymmetric key operations of cloud users and CSPs; it provides a level of mutual verifiability that is identical to that of a PKI; it also minimizes billing transaction latency.
- **Support for a mutually verifiable billing mechanism:** In traditional cloud billing systems, the billing management and billing information are both processed by the CSP alone. For a credible and verifiable way of logging resource usage, a digital signature is essential because it enhances the billing mechanism with mutual verifiability [2]. Our proposed billing system, THEMIS, uses the novel cloud notary authority to generate mutually verifiable binding information for users and CSPs. Furthermore, the resource usage log, which is based on a one-way hash chain, retains the information in its local storage for future accusations. THEMIS supervises billing

[1] http://www.icubecloud.com:Cloud service platform & testbed of this work

978-0-7695-4130-3/10 $26.00 © 2010 IEEE
DOI 10.1109/CLOUD.2010.21

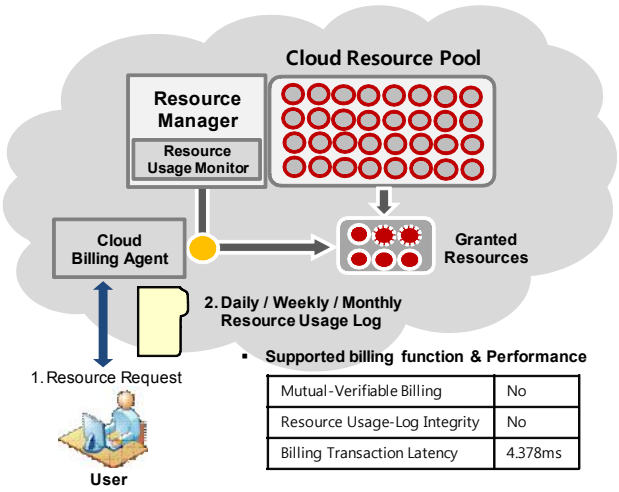


Figure 1. Native billing system and a brief overview of the characteristics

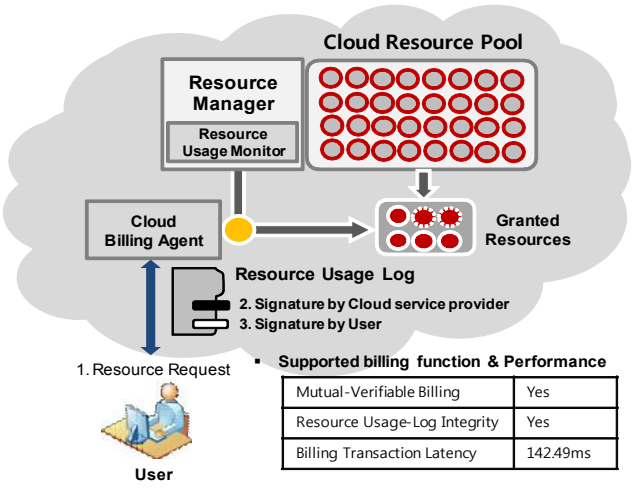


Figure 2. Billing system enhanced with PKI and its characteristics

and, because of its objectivity, is likely to be accepted by users and CSPs alike.

In this paper, we present our realized THEMIS billing system; it meets the above requirements and is deployed in a real cloud computing service. According to the performance evaluation, the overall latency of the billing transactions of THEMIS (4.606 ms) is much shorter than the latency of PKI-based billing transactions (which averages 142.49 ms), though THEMIS provides identical security features as a PKI.

The remainder of the paper is organized as follows: In Section 2, we discuss relevant works. In Section 3, we present the overall system design and components of the proposed billing system. In Section 4, we illustrate the proposed billing protocol. In Section 5, we evaluate the performance of the proposed billing scheme. Finally, in Section 6, we present our conclusions and future works.

2. Previous Work

A billing system that tracks the usage of computing resources has been actively studied and developed in the research area of grid or cloud computing. To date, however, none of the billing systems have incorporated a verifiable or fine-grained billing mechanism. In this section, we briefly discuss the experimental results as we evaluate existing billing systems in terms of their security level and billing overhead. A more comprehensive evaluation of the experimental results is described in Section V.

2.1. Native billing systems

In a pay-as-you-go pricing model, users can scale the capacity of cloud resources on demand. Resource consumption is billed on a utility basis with little or no upfront cost. Two pioneering studies identified challenges in managing the resources of a grid computing environment and proposed a computational economy as a metaphor for effective management of resources [4, 5]. Several researchers presented a resource usage processing system that can be used to scan batch system logs to build accounting records [6-10, 21]; this system is able to record and account for the usage of grid resources. Figure 1 shows the architecture and characteristics of a billing system without any security concerns. As shown in Figure 1, the resource usage information, such as the CPU cycles, storage, and network bandwidth, are collected via a resource monitor and charged over the billing module. APEL [6] presents a billing system that processes log information to create quantified accounting records. There are also other resource management and billing frameworks that have been suggested as part of traditional grid approaches: namely, Condor/G [7], Nimrod/G [8], GRASP [9], Tivoli [10], and TeraGrid [21]. However, rather than address security concerns, those frameworks mainly focus on presenting distributed resource usage metering as well as an accounting and account balancing mechanism for a distributed grid environment. Thus, they fail to provide the type of mutual verifiability and integrity needed in a verifiable billing system that records the usage of cloud resources.

2.2. Security-enhanced billing systems

Several electronic payment schemes have been proposed in the literature in an attempt to provide security-enhanced billing mechanisms. They include micropayment-based schemes such as PayWord [18], MiniPay [16], e-coupons [19], and NetPay [24]. Broadly deployed in e-payment systems, these schemes enable users to securely and efficiently perform repeated

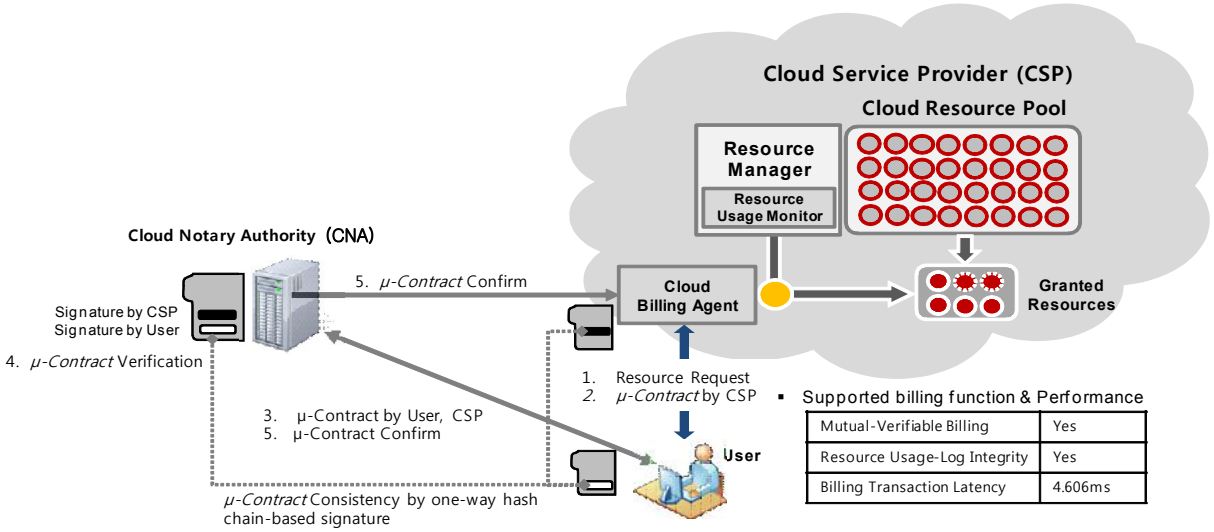


Figure 3. Overall architecture and process flow of THEMIS

payments. Many of these schemes are based on the use of one-way hash functions that generate chains of hash values; users perform billing transactions by releasing a certain number of hashes in the hash chain. Although these schemes are supposed to provide secure billing for micropayment transactions in a computationally efficient way, they offer no support for mutual verifiability of the resource usage log.

The research area of cloud or grid computing has presented the following market models and PKI enhanced billing and accounting frameworks for user authentication and secure communication: RUS [10], DGAS [11], SGAS [12], GridBank [13], and Amazon EC2 [14]. They have an access control mechanism that uses digitally signed certificates to define and enforce an access policy for a set of distributed resources. However, none of these projects addresses the problem of dealing with the verifiability of billing transactions or its computational efficiency, as we do for the cloud computing environment.

Due to its digital signature and non-repudiation features, a PKI is generally considered to be the most appropriate and fundamental way of enforcing verifiability in terms of our requirements [15, 25]. Figure 2 illustrates the organization of a PKI-enhanced billing system and its characteristics in terms of security level and billing overhead. A PKI has a billing overhead even though it relies on full security features to achieve mutual verifiability. The extent of the overhead is mainly determined by the extremely high complexity of RSA operations when the PKI is used for a billing system by a thin client or heavily loaded server [20]. Figure 2 shows that the estimated average billing transaction latency in our experimental environment is about 142.49 ms.

3. THEMIS† Architecture

While deliberating on the system requirements mentioned above, we based the design of our billing system and protocol on two principles: verifiable billing with a cloud notary authority; and computationally efficient billing transactions. In this section, we present the overall architecture and billing process of THEMIS.

3.1. The Proposed THEMIS Infrastructure

Figure 3 shows the overall architecture of our THEMIS billing infrastructure. The three major components of the architecture are listed as follows:

- **Cloud Service Provider (CSP):** The CSP enables users to scale their capacity upwards or downwards in accordance with their computing requirements and to pay only for the capacity that they actually use.
- **Users:** We assume that users are thins clients who use services in the cloud computing environment. To use services in such an environment, each user makes a request to the CSP with a billing transaction.
- **Cloud Notary Authority (CNA):** The cloud notary authority provides a mutually verifiable integrity mechanism to combat the malicious behavior of users or the CSP. The cloud notary authority investigates billing transactions and generates mutually verifiable binding information

† THEMIS, Goddess of Justice: We named our system to THEMIS because we are pursuing the justice of the Cloud.

Table 1. Notations of the entities and messages

Definition of the entity symbols
c: Cloud Service Provider (CSP) u: User n: Cloud-Notary-Authority
Definition of the Message symbols
$K_{\alpha,\beta}$: Shared Key between a and b PK_{α} : Public Key of a SK_{α} : Private Key of a H(M) : Hash result for message M Usage Sequence Hash Chain: $C_0 \rightarrow C_1 \rightarrow C_2 \rightarrow C_3 \rightarrow \ldots \rightarrow C_n$, $H(C_n) = C_{n-1}$ Hash Generation Sequence $C_{\alpha,n}$: n^{th} value of the α's hash chain T_s : Time-Stamp N_a: Random value for preventing replay attack by a { M }K : M encrypted by K { M }PK : M encrypted by public Key { M }SK : Digital signature for M by private key

among all the involved entities on the basis of a one-way hash chain and resource usage log; it also retains the information in its local storage for future accusations. Due to the cloud notary authority, our proposed billing system can provide mutual verifiability and integrity for a cloud resource usage log. The process, which involves exclusive key sharing between entities and a one-way hash chain-based signature, is computationally efficient for a thin client and the CSP.

3.2. Overall Billing Process of the THEMIS

With the proposed cloud notary authority, mutually verifiable billing can be provided without asymmetric key operations of any entities after a renewal phase. Figure 3 shows the overall process of a billing transaction with our billing system. The following details of the mutually verifiable billing mechanism are described in more detail in Section 4:

1. The user generates a cloud resource request message and sends it to the CSP.
2. The CSP sends the user a μ-contract-CSP message generated with a digital signature from a CSP hash chain.
3. The user generates a μ-contract with a hash chain-based digital signature of the user and sends it to the cloud notary authority.
4. The cloud notary authority performs transactions to verify the μ-contract from the user.
5. The billing process is completed when the user and the CSP receive confirmation from the cloud notary authority.

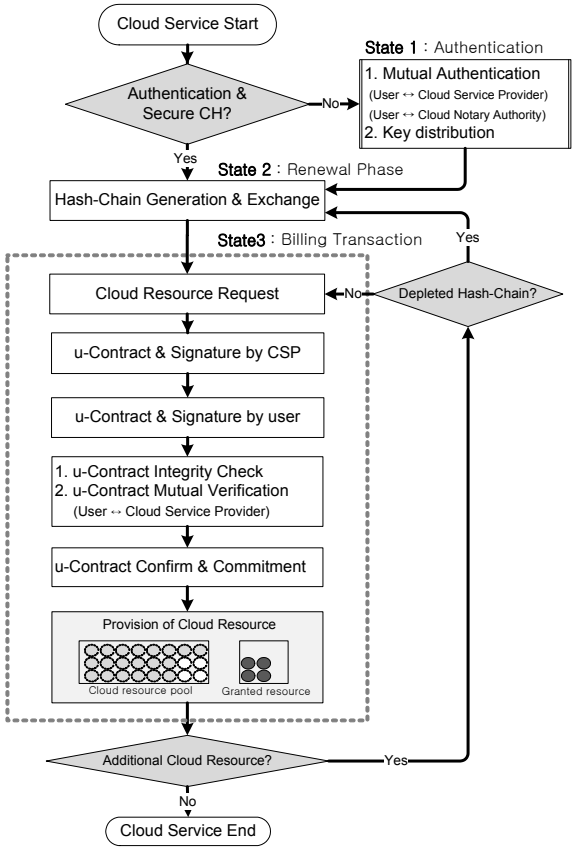


Figure 4. Flow diagram of the proposed billing protocol

4. Proposed Billing Protocol

While deliberating on our system design philosophy, we did our utmost to streamline the computation and communication overhead of the billing operation. Our novel *μ-contract* is generated by a hash function and it exclusively distributes keys among entities. It can optimize the computation and communication overheads of the billing mechanism and facilitates mutual verifiability and integrity for cloud resource usage. In this section, we describe the overall transactions of the proposed billing protocol. Table 1 describes the notations of the entities and messages that describe the proposed protocol.

4.1. Flow Diagram of the Billing Protocol

Figure 4 shows a flow diagram to present the transactions of the proposed billing protocol. Figure 5 describes the overall billing transactions and the message specification that describe the proposed protocol. The protocol consists of three states and the protocol safety is verified in Section 4.4:

- ***State 1*:** When a user first accesses the CSP, mutual PKI-based authentications are performed by the user,

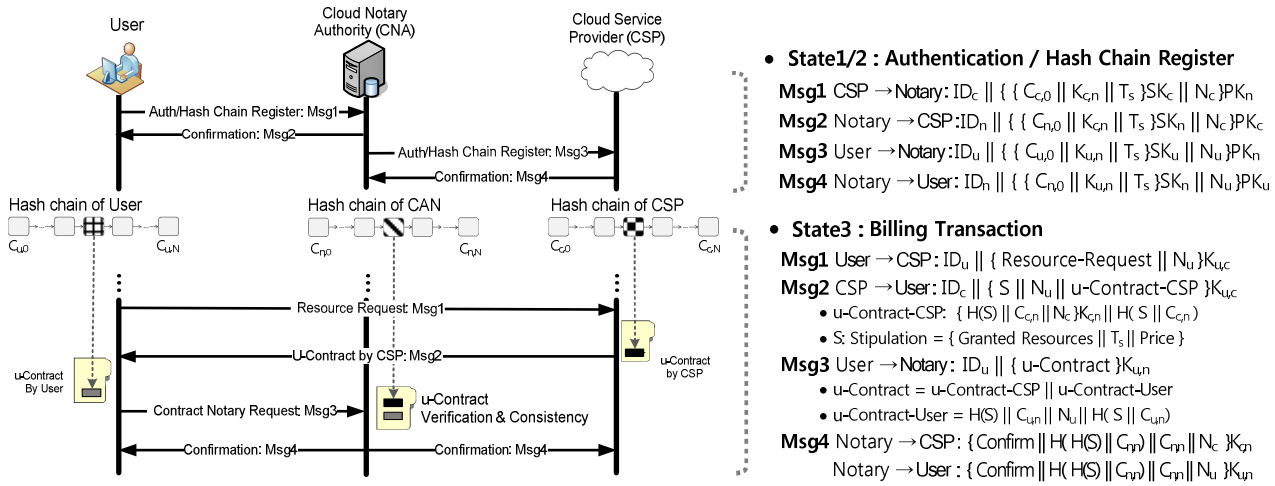


Figure 5. Overall billing transactions of THEMIS

the CSP, and the cloud notary authority. Throughout the mutual authentications, the following three keys are shared exclusively among the user, the cloud notary authority, and the CSP:

- *User ↔ CSP:* $K_{u,c}$
- *User ↔ CNA:* $K_{u,n}$
- *CSP ↔ CNA:* $K_{c,n}$

- ***State 2:*** In this state, the user, the CSP, and the CNA generate a hash chain of length N by applying the hash function N times to a random value ($C_{u,N}$, $C_{c,N}$) so that a final hash ($C_{u,0}$, $C_{c,0}$) can be obtained. The user and the CSP commit to the hash chain by digitally signing the final hash ($C_{u,0}$, $C_{c,0}$), and by registering them and the keys ($K_{u,n}$, $K_{c,n}$). As shown in Figure 5, the purpose of this registration is to share with the cloud notary authority and to receive the hash chain ($C_{n,0}$) generated by the cloud notary authority. Once the commitment of the one-way hash chains is successfully completed, *State 2* is skipped until the corresponding hash chain either expires or is revoked.
- ***State 3:*** A user who intends to receive a cloud resource from the CSP sends a resource request message and then receives a *μ-contract* message from the CSP. The user generates a *μ-contract* by using the symmetric key operation and hash function and then transmits the contract to the cloud notary authority. When the *μ-contract* is received from the user, the cloud notary authority verifies the contract and sends confirmation messages to both the user and the CSP. When the confirmation message is received from the cloud notary authority, the user and the CSP confirm the contract between the user and CSP and the billing operation is terminated.

4.2. Description of THEMIS Billing Protocol

As previously mentioned, the proposed protocol can provide a verifiable and non-obstructive method of billing after the authentication (*State1*) and the hash chain registration or renewal (*State2*). Any user who accesses the CSP for the first time or the renewal time is asked to generate a hash chain and commit to it by digitally signing the final hash, which requires a private key operation. Hence, the initial billing operation takes longer than a normal PKI-based billing operation. On the other hand, the billing overhead for the client and the CSP can be reduced drastically after the completion of *State 2* because the user and the CSP can perform the billing operation by using a much simpler type of symmetric key operation and hash operation.

A user who intends to receive a cloud resource from the CSP generates a cloud request message (*Msg1*) encrypted with $K_{u,c}$ and sends it to the CSP. When the message is received, the CSP transmits a stipulation containing an agreement that covers factors such as the granted resource, the time, and the price as well as the *μ-contract-CSP*. The *μ-contract-CSP* contains an encrypted value of three inputs (namely, a hashed value of the stipulation (S), a hash element ($C_{c,n}$), and a randomly generated nonce) and a hashed value of two inputs (namely, a stipulation (S) and a hash element ($C_{c,n}$)). The hash element is updated for each billing transaction on a chain-by-chain basis so that all of the hash elements can be linked and verified sequentially toward the seed value ($C_{c,0}$) of the hash chain. Furthermore, the used hash element ($C_{c,n}$) is unknown to the user. When the user receives the *μ-contract-CSP* (*Msg2*), the user generates a billing request message (*Msg3*) by combining the *μ-contract-CSP* from the CSP with the user's own *μ-contract-User*. The user then sends *Msg3* to the cloud notary authority for the mutual verifiability and integrity of the *μ-contract*. When the message arrives, the cloud notary authority checks the validity of the *μ-contract* by

comparing the *H(S)* section of *μ-contract-User* with the *H(S)* section of *μ-contract-CSP*. The user and the CSP should have an identical stipulation (*S*). The cloud notary authority then sends the user and the CSP the verification result (*Msg4)*, which contains the hashed value of *H(S)* and the hash element ($C_{n,n}$). The user and the CSP can subsequently verify the integrity of *H(S)* in the *μ-contract* by using $C_{n,n}$. The billing transaction is completed when the user and the CSP receive the final confirmation message.

4.3. How to prove the user billing records

Our proposed protocol can provide mutual verifiability and integrity through the *μ-contract*, which is generated among the entities by symmetric and hash cryptography and distributed keys ($K_{u,c}$, $K_{u,n}$, $K_{c,n}$). This section elaborates how billing can be verified in collaboration with the cloud notary authority. Mutually verifiable billing can be achieved when the cloud notary authority verifies the validity of the user's message for each billing transaction in *State3*. The transaction does not require any asymmetric key operations of the user and the CSP. Throughout the verification phase (*Msg3, Msg4*), the cloud notary authority generates and retains binding information which states that the CSP sent *Msg2* and that the user sent *Msg3* with the same stipulation (*S*). This process is a type of notarized billing list (NBL). The NBL is a data structure for storing evidence of the billing transactions for future accusations. All of the contexts are periodically stored with the digital signature of the cloud notary authority to ensure the integrity of the NBL context. In addition, the NBL can provide mutual verifiability and integrity because all the transactions are linked to each other with the hash chains of the user, the CSP, and the cloud notary authority. Let's assume that the CSP asserts that the user repudiates a certain CSP billing. In this case, the CSP can submit to the cloud notary authority a claim for justice, drawing attention to the hash element ($C_{c,n}$) and stipulation (*S*) included in *μ-contract-CSP*. The cloud notary authority then demands to see the stipulation (*S*) used to generate the *μ-contract-User* together with the hash element ($C_{u,n}$) and inputs them to the hash function. Any discrepancy between the output of the hash function and the stored data of the cloud notary authority proves that the user has modified the stipulation of the relevant billing. The cloud notary authority can therefore refute the user's repudiation.

4.4. Proof of the Security of THEMIS

Our analysis of the protocol safety is based on consideration of the replay attacks and the man-in-the-middle (MITM) attacks. We assumed that the underlying cryptography (AES, RSA, and SHA)[3] was invulnerable with regard to message secrecy and integrity; hence, we ignored attacks such as cryptanalysis and message slicing. One the other hand, any principle can place or inject a message on any link at any time. In addition, any principle can see, delete, alter, and redirect all exchanged messages being passed along any link or replay messages recorded from past communications. In this section, we confirm that our proposed protocol is safe from replay attacks and MITM attacks.

Theorem1. THEMIS is safe from a malicious CSP or malicious user.

Proof: We prove the safety for the next attack types as follows:

1) Forgery of *μ-contract* before the provision of cloud resources by a malicious CSP or a malicious user.
2) Forgery of *μ-contract* after the provision of cloud resources by a malicious CSP or a malicious user.

- **In case of 1)**, a malicious CSP or a malicious user falsifies the *μ-contract* before any cloud resources are provided. However, the forgery by a malicious CSP or a malicious user cannot succeed because the cloud notary authority checks the validity of *μ-contracts* from the CSP and the user. The cloud notary authority can compare the hash element in the CSP's *μ-contract-CSP* with that in the user's *μ-contract-User*. If these elements don't matched, the cloud notary authority does not send a confirm message to the CSP and the user. Thus, neither the CSP nor user can successfully falsify the *μ-contract* before the provision of cloud resources.
- **In case of 2)**, a malicious CSP or a malicious user falsifies the *μ-contract* after the provision of cloud resources. However, the falsification of the malicious CSP or the malicious user cannot succeed because the cloud notary authority already knows the verifiable *μ-contracts* of both the CSP and the user. Hence, the cloud notary authority can identify the malicious provider or user by comparing the *H(S)* and C_n sections in the *μ-contracts* of the cloud notary authority with those in the *μ-contracts* of the CSP and the user. □

Theorem2. THEMIS is safe from replay attacks.

Proof: Let's assume that the billing path is [User ↔ Cloud-Notary-Authority ↔ CSP, User]. We prove the safety for the next attack types as follows:

[3] **AES(Advanced Encryption Standard)** is an encryption standard on the basis of symmetric key operation (FIPS 197).
RSA(Rivest, Shamir, and Adleman) is an algorithm for asymmetric -key cryptography.
SHA(Secure Hash Algorithm) is one of a number of cryptographic hash functions published by the National Institute of Standards and Technology.

1) A replay attack for a resource request message (*Msg1*) and *μ-contract* messages (*Msg2* and *Msg3*).
2) A replay attack for the confirmation message (*Msg4*).

- **In case of 1)**, an intruder can try to attack by sending the captured messages. However, the intruder's attack cannot succeed because the messages include the nonce data (N_u) and a *μ-contract* that is altered with a time stamp for each billing transaction.
- **In case of 2)**, an intruder can try to attack by sending the captured messages. However, the intruder's attack cannot succeed because the messages include the nonce data (N_u, N_c, and N_n). □

Theorem3. THEMIS is safe from MITM attacks.
Proof: We prove the safety for the next attack types as follows:
1) MITM attack between a user and a CSP
2) MITM attack between a cloud-notary-authority and a user.
3) MITM attack between a cloud-notary-authority and a CSP

- **In case of 1)**, the CSP and user generate billing transaction messages that are altered for each billing transaction, and the messages are encrypted and transmitted by means of the shared keys that are paired in a previous state. Thus, the intruder cannot successfully masquerade as the user or the service device.
- **In case of 2) and 3)**, each entity performs a mutual authentication over the PKI and shares the key for a secure connection. Thus, the intruder cannot forge the message of the cloud-notary authority. □

5. Efficiency and Performance Evaluation

In this section, we present the performance results of our prototype version of THEMIS. First, we demonstrate the overall experimental environment. We then describe the operational efficiency of the billing protocol to evaluate the performance of THEMIS in terms of billing overhead.

5.1. Experimental Environment

To evaluate the performance characteristics of THEMIS, we constructed a cloud user emulator and coupled it to a billing transaction generator. The objective of the emulator is to simulate the processing and communications resources anticipated in a full implementation. Figure 6 shows the overall experimental environment. The operating times of the emulator are similar to actual operating times (for communications, billing operations, and message processing). In addition, a user emulator is connected to THEMIS and the emulator receives control signals from the billing transaction generator. The generator is a module that generates control signals to produce billing request messages; for this purpose, we use a random generator that models the computer usage pattern of users [22].

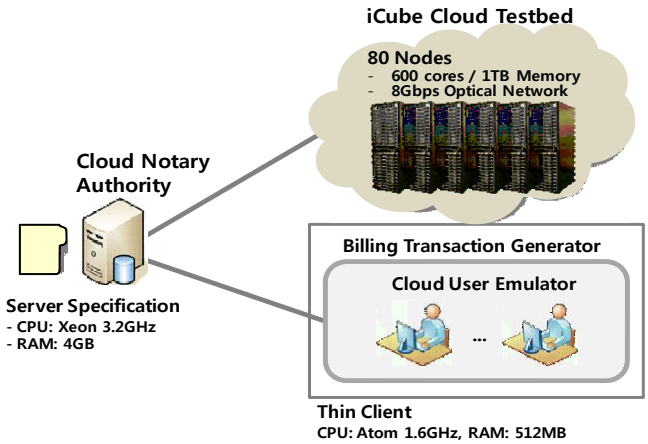
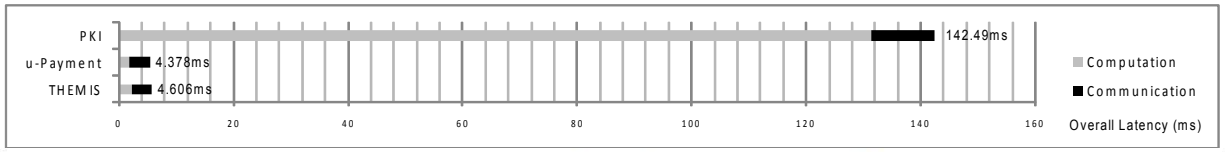


Figure 6. Overall experiment environment

5.2. Billing Protocol Efficiency and Comparative Evaluation

The performance of the billing protocol in terms of billing overhead and the consumption of processing and communication resources is an important factor to be considered when designing billing protocols. First, we analyze how the efficiency of THEMIS compares with PKI-based billing; we also analyze the micropayment in terms of computation and communication efficiency.

Figure 7 gives the number of public and private keys (RSA 1024 bit), the symmetric key (AES 128 bit), and the hash (SHA-1) operations performed with the total operating time for each billing protocol. In spite of its smaller number of cryptography operations per billing, the PKI-based billing protocol has a much longer latency of billing transactions than other schemes because it has a certain number of private and public key operations for all of the entities. The micropayment, on the other hand, has the shortest operating time because it can be completed by symmetric key and hash operations but it fails to meet our security requirement. In the case of THEMIS, the first time a user accesses the CSP, all of the entities perform mutual PKI-based authentications and generate a hash chain which requires two private and public key operations and multiple hash operations. The authentication and hash chain generation time of THEMIS is similar to the operating time of PKI-based billing. However, after the operations, the user can perform a billing operation by processing only four symmetric key operations and two hash operations; this process has a much shorter operating time than the corresponding process of a PKI-based billing transaction. By way of summarizing the above results, we give an outline of the overall transaction time of the billing protocols. The PKI-



System	Security Functionality		User				Cloud Service Provider (CSP)				Notary Authority (Trust 3rd Party)				Communication	Total Transaction Time (Computation + Communication)
	Mutual Verifiability	Integrity	RSA		AES	SHA	RSA		AES	SHA	RSA		AES	SHA		
			PU	PR			PU	PR			PU	PR				
Native PKI-based Billing	Yes	Yes	2	1	3	0	1	2	3	0	1	0	0	0	Avg.11.08ms	Avg.142.49ms
			Avg. 125.21 ms				Avg. 6.062 ms				Avg. 0.136 ms					
micro-Payment	No	No	0	0	4	0	0	0	0	0	0	0	4	1	Avg. 3.44ms	Avg. 4.378ms
			Avg. 0.912 ms				Avg. 0.0 ms				Avg. 0.026 ms					
THEMIS	Yes	Yes	0	0	4	2	0	0	3	1	0	0	4	2	Avg. 3.58ms	Avg. 4.606ms
			Avg. 0.982 ms				Avg. 0.028 ms				Avg. 0.016 ms					

PU: Public Key Operation, PR: Private Key Operation

Figure 7. Total transaction overhead for each billing protocol

based billing transaction time with the RSA 1024 bit algorithm is 142.41 ms. In the case of THEMIS, the billing transaction can be accomplished without asymmetric key operations. These results confirm that THEMIS has a much shorter operating time, in spite of its mutual verifiability and integrity. As shown in Figure 7, we estimated the operating time of each entity for each billing transaction so that we could measure how much cryptography causes an obstructive billing overhead. We measured the billing transaction latency by measuring the time interval between the start time of *Msg1* and the end time of *Msg4* on the client side. With THEMIS, the total transaction time for the user and the CSP is much shorter than that of the PKI-based billing system. Moreover, without compromising the PKI security level, THEMIS's total billing transaction time (4.606 ms) is much shorter than that of PKI-based billing (142.49 ms).

5.3. Storage Overhead

The cloud notary authority needs to store all the *μ-contracts* contained in the messages it receives so that mutual verifiability can be ensured for an indefinite period. First, we measured the size of the *μ-contracts* and the related index stored in the *iCube Cloud Testbed* by the cloud notary authority, the users, and the CSP for a period of one month. The *iCube Cloud Testbed* is currently used by three research institutes (Seoul National University, the University of Paris VI, and the Korea Advanced Institute of Science and Technology) for course work and research activities. The total number of users is about 300. The average size of the *μ-contracts* stored by the cloud notary authority per user per day is 38 KB, and the average size of the usage log and related information stored on by the cloud and the user is 875 KB per user per day. If a million people use the cloud service, THEMIS will need archiving capabilities so that after a certain period records can be moved from the cloud notary authority into archival storage. Currently, we are working towards the scalability and optimization of the storage requirements for THEMIS billing transactions.

6. Conclusion and Future work

Our aim was to provide a full-fledged verifiable and non-obstructive billing solution tailored for a cloud computing environment. To accomplish this task, we thoroughly reviewed the ways in which conventional billing systems are used in the environment, and we consequently derived blueprints for our mutually verifiable and computationally efficient billing system called THEMIS. Besides utilizing conventional billing systems, we conceived and implemented the concept of a cloud notary authority that supervises billing to make it more objective and acceptable to users and CSPs alike. THEMIS ensures undeniable verification of any transaction between a user and a CSP. Furthermore, our mutually verifiable billing protocol significantly reduces the billing overhead. Our next step is to consider the scalability and fault tolerance of THEMIS. Currently, we are investigating THEMIS from the perspectives of massive scalability and robustness. We believe that putting multiple trusted third parties in charge of the cloud notary authority is an appropriate way forward, as is the case with the PKI. We are working towards a THEMIS-based system with more fault tolerance to scalable billing.

7. Acknowledgments

The authors wish to thank their anonymous referees for all of their invaluable comments and suggestions. They would also like to NexR Co. Ltd for their help in conducting some of the experiments in this study. The work presented in this paper was supported by MKE

(Ministry of Knowledge Economy, Republic of Korea), Project No. A1100-0901-1710.

8. References

[1] Ristenpart T., Tromer E., Shacham H, and Savage S., "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds", 16th ACM Conference on Computer and Communication Security (CCS) 2009.

[2] H.-Y. Lin, and L. Ham, "Authentication protocols with non-repudiation services in personal communication system", IEEE Communications Letters 3 (8). 1999.

[3] Azure Service, "Microsoft, Windows Azure Platform", http://www.microsoft.com/windowsazure/

[4] Buyya R, Abramson D, Giddy J, Stockinger H. "Economic models for resource management and scheduling in Grid computing", Concurrency and Computation: Practice and Experience 2002; 14(13–15):1507–1542.

[5] Chun BN, Culler DE. Market-based proportional resource sharing for clusters. Technical Report CSD-1092, Computer Science Division, UC at Berkeley, January 2000.

[6] R Byrom , "APEL: An implementation of Grid accounting using R-GMA, et al.", 2005 UK e-Science All Hands Meeting, September 2005

[7] M. Litzkow, M. Livny, and M. Mutka, "Condor: A hunter of idle workstations", 8th Int. Conf. Distributed Computing Systems (ICDCS 1988), San Jose, CA, Jan. 1988.

[8] Buyya R, Abramson D, Giddy J., "Nimrod/G: An architecture for a resource management and scheduling system in a global Computational Grid", Proc. of the 4th International Conference on HPC, May 2000.

[9] Kwon, O., Hahm, J., Kim, S., and Lee, J. 2004., "GRASP: A Grid Resource Allocation System based on OGSA", In Proc. of the 13th IEEE international Symposium on High Performance Distributed Computing.

[10] IBM, "Tivoli: Usage and Accounting Manager", IBM Press 2009

[11] Guarise, A., Piro, R., and Werbrouck, A. "DataGrid Accounting System - Architecture", EU DataGrid, 2003.

[12] P. Gardfj¨all, E. Elmroth, L. Johnsson, O. Mulmo, and T. Sandholm. Scalable grid-wide capacity allocation with the SweGrid Accounting System (SGAS). Concurrency Computat.: Pract. Exper., 20(18):2089–2122, 2008.

[13] A. Barmouta and R. Buyya, "GridBank: A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration", Proceedings of the 17th IEEE IPDPS 2003, April,2003, pp. 22–26.

[14] Amazon Web Services, "Amazon Elastic Compute Cloud", http://aws.amazon.com/ec2/

[15] Thompson, M., Johnston, W., Mudumbai, S., Hoo, G., Jackson, K., and Essiari, "Certificate-based access control for widely distributed resources". In Proceedings of the 8th Conference on USENIX Security Symposium

[16] A. Herzberg, H. Yochai, "Mini-pay: charging per click on the web", in: 6th World Wide Web Conference, Santa Clara, USA, April 1997.

[17] Stefan Kelm, "Public Key Infrastructure", http://www.pki-page.org/ ,2009

[18] R. Rivest, A. Shamir, "PayWord and MicroMint: two simple micropayment schemes", 1996 International Workshop on Security Protocols, Lecture Notes in Computer Science, vol. 1189, Springer, pp. 69–87

[19] V. Patil, R.K. Shyamasundar, "An efficient, secure and delegable micro-payment system", 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service, Taipei, Taiwan, 28–31 March 2004.

[20] J. Jonsson and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", Network Group

[21] E. Roberts, M. Dahan, J. Boisseau. "TeraGrid User Portal: An Integrated Interface for TeraGrid User Information and Services", TeraGrid 2008

[22] D.Banks, J.S.Erickson, and M.Rhodes, "Towards Cloud-based Collaboration Services", Usenix Workshop HotCloud 2009

[23] N.Santos, K.P.Gummadi, and R.Rodrigues, "Towards Trusted Cloud Computing", Usenix Workshop HotCloud 2009

[24] Dai, X., and Grundy, J. "NetPay: an off-line, decentralized micro-payment system for thin-client applications", E-Commerce Research and Applications, 6, 2007,91–101

[25] Park, K., Lim, S. S., and Park, K. H., "Computationally Efficient PKI-Based Single Sign-On Protocol, PKASSO for Mobile Devices", IEEE Trans. Computers. 57, 6 (Jun. 2008), 821-834