THEMIS: A Mutually Verifiable Billing System

for the Cloud Computing Environment

**Abstract**

With the widespread adoption of cloud computing, the ability to record and account for the usage of cloud resources in a credible and verifiable way has become critical for cloud service providers and users alike. The success of such a billing system depends on several factors: the billing transactions must have integrity and no repudiation capabilities; the billing transactions must be non obstructive and have a minimal computation cost; and the service level agreement (SLA) monitoring should be provided in a trusted manner. Existing billing systems are limited in terms of security capabilities or computational overhead. In this paper, we propose a secure and non obstructive billing system called THEMIS as a remedy for these limitations. The system uses a novel concept of a cloud notary authority for the supervision of billing. The cloud notary authority generates mutually verifiable binding information that can be used to resolve future disputes between a user and a cloud service provider in a computationally efficient way. Furthermore, to provide a forgery-resistive SLA monitoring mechanism, we devised a SLA

monitoring module enhanced with a trusted platform module (TPM), called S-Mon. The performance evaluation confirms that the overall latency of THEMIS billing transactions (avg. 4.89 ms) is much shorter than the latency of public key infrastructure (PKI)-based billing transactions (avg. 82.51 ms), though THEMIS guarantees identical security features as a PKI. This work has been undertaken on a real cloud computing service called iCubeCloud.

**SYSTEM ANALYSIS**

**EXISTING SYSTEM**

The billing systems with limited security concerns and the micropayment-based billing system require a relatively low level of computational complexity: the nonobstructive billing transaction latency is 4.06 ms for the former and 4.70 ms for the latter. Nevertheless, these systems are inadequate in terms of transaction integrity, nonrepudiation, and trusted SLA monitoring. In spite of the consensus that PKI-based billing systems offer a high level of security through two security functions (excluding trustworthy SLA monitoring),the security comes at the price of extremely complex PKI operations. Consequently, when a PKI-based billing system is used in a cloud computing environment, the high computational complexity causes high deployment costs and a high operational overhead because the PKI operations must be performed by the user and the CSP.

**DRAWBACK IN EXISTING SYSTEM**

* Existing billing systems are limited in terms of security capabilities or computational overhead.
* Maximum Computation Cost
* No Trusted SLA Monitoring

**EXISTING ALGORITHM**

Public key infrastructure (PKI)-based billing transactions

**PROPOSED SYSTEM**

In this paper, we propose a secure and nonobstructive billing system called THEMIS as a remedy for these limitations. The system uses a novel concept of a cloud notary authority for the supervision of billing. The cloud notary authority generates mutually verifiable binding information that can be used to resolve future disputes between a user and a cloud service provider in a computationally efficient way.

**ADVANTAGES IN PROPOSED SYSTEM**

* The billing transactions must be non obstructive.
* Minimal Computation Cost
* Service level agreement (SLA) monitoring should be provided in a trusted manner.
* Faster time to market
* Accurate, consistent and competitive pricing

**PROPOSED TECHNIQUE**

SLA monitoring mechanism

**SYSTEM REQUIREMENTS**

**HARDWARE**

PROCESSOR : PENTIUM IV 2.6 GHz, Intel Core 2 Duo.

RAM : 512 MB DD RAM

MONITOR : 15” COLOR

HARD DISK : 40 GB

**SOFTWARE**

Front End : J2EE (JSP, SERVLET)

Back End : MS SQL 05

Operating System : Windows 07

IDE : Net Beans, Eclipse

**FUTURE ENHANCEMENT**

We haveEnhance a customer service to get the feedback for that the billing service in cloud and analyse the performance of cloud in that billing service. Improving the availability of services in cloud.