2012 19th Asia-Pacific Software Engineering Conference

SaaS Application Framework using Information Gateway Enabling Cloud Service with Data Confidentiality

Kiyoshi Nishikawa, Kenji Oki and Akihiko Matsuo
Software Systems Laboratories,
Fujitsu Laboratories Ltd., Kawasaki, Japan
Email: {nishikawa.kiyos, oki.kenji, a_matsuo}@jp.fujitsu.com

***Abstract*—Along with the rapid evolution of cloud computing, outsourcing service has also changed significantly. One of the main concerns for cloud computing is security. In particular, it becomes more important to handle client data securely in such a consulting service. In this paper, we present the design and implementation of Software as a Service (SaaS) application framework using Information Gateway that enables cloud service while maintaining data confidentiality. By setting up Information Gateway in the client environment, the executing location is dynamically controlled according to whether the data contains confidential information or not, and only secured data is routed to the SaaS application in the cloud. Therefore, if the data policy is registered by the client administrator beforehand, the user is able to use the cloud service appropriately without being aware of it. Moreover, we also describe that existing applications can be easily ported into the SaaS application because our framework allows application developers to define complicated routing logic briefly.**

***Keywords*-cloud computing; client data; SaaS application; Information Gateway; integration framework; sandbox; security manager; existing applications**

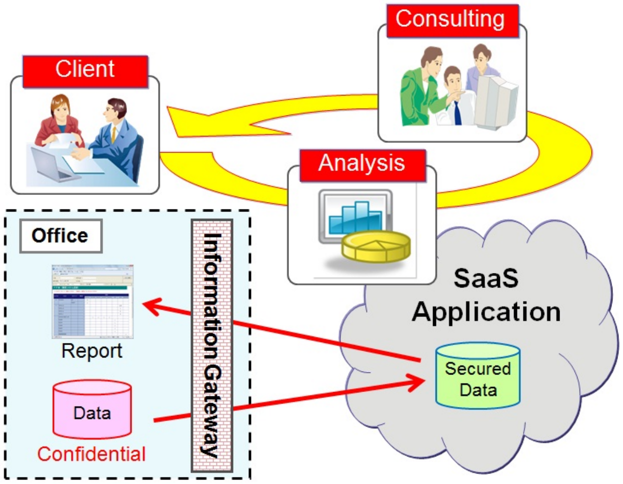


Figure 1. Proposing Consulting Service Image

I. Introduction

Cloud computing has recently emerged as a new paradigm shift for hosting and delivering services [1] [2]. The maximum feature of cloud computing is on-demand service delivery. The range of services includes the Infrastructure, Platform and Software [3] [4]. Running applications remotely in cloud computing has a number of advantages, such as reducing Information and Communication Technology (ICT) cost, flexibly scaling computing resources, and choosing services freely according to the demand. So it gets attention from many enterprises. However, one of main concerns for cloud computing is security. This security issue prevents many enterprises from adopting cloud computing for the purpose of business.

In such an ICT consulting service, in order to achieve client satisfaction, the outsourcing vendor analyzes the current ICT situation, and delivers expert insight and robust recommendations that strengthen client's decision-making, in alignment with business strategies. A trusted relationship between the client and the outsourcing vendor is most important for the consulting business. As a rule, since an ICT system related to the business contains a lot of confidential and privacy information, the outsourcing vendor should pay attention to handling client data, as set forth in the contractual agreement. Thus, the outsourcing service need not necessarily be an on-demand delivery like cloud computing [5].

In this paper, we present the design and implementation of SaaS application framework using *Information Gateway* that enables the use of SaaS application in the cloud with client data confidentiality. In addition, we also discuss the result of verifying our framework from the viewpoint of integration solution and application developers.

II. Information Gateway

Until now, in order to use the SaaS application, all the client data had to be stored in the cloud. For example, when client data contained confidential information, it was not possible to use the SaaS application. Our target is to enable the use of SaaS application without sending confidential information to the cloud, as illustrated in Figure 1. By setting up Information Gateway in the client environment beforehand, the client user is able to use the SaaS application securely without being conscious of data policy. The functions required for the Information Gateway are as follows.

1530-1362/12 $26.00 © 2012 IEEE
DOI 10.1109/APSEC.2012.133

A. Data Encryption (Decryption)

The simple case is that Information Gateway encrypts the confidential information based on a data policy defined beforehand, and encrypted data is routed to a SaaS application in the cloud, as illustrated in Figure 2. As a result, the client user is able to use the SaaS application without sending confidential information to the cloud. The information on which item was encrypted is stored in the Information Gateway. On the other hand, if the client user receives some processing results from the cloud, the Information Gateway decrypts the results again, and returns it to the user. Thus, Information Gateway needs a component to manage the data policy instead of the user.

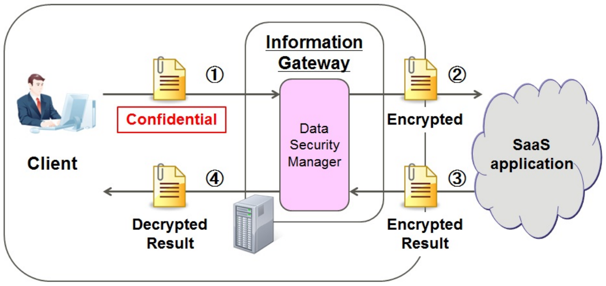


Figure 2. Data Encryption (Decryption)

B. Dynamic Determination of Executing Location

However, if data is encrypted first, there are some cases that SaaS application is not available for the encrypted data. In such a case, as illustrated in Figure 3, the Information Gateway should be able to dynamically switch the executing location to the client environment based on both the data policy and the attribute of SaaS application, and then pre-processing should be executed securely in the client environment, using a deploying sandbox. After pre-processing, only secured data must be routed to the SaaS application in the cloud. Thus, Information Gateway also needs a component to manage the attribute of SaaS application so as to determine where a part of the processing must be executed.

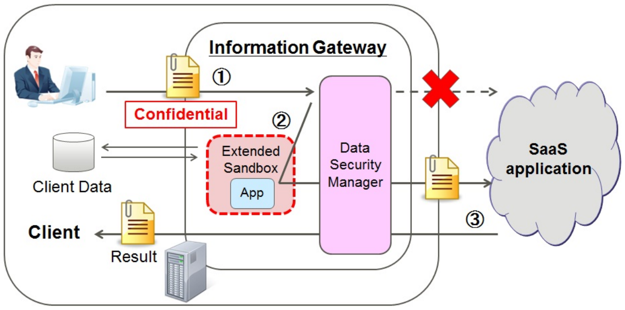


Figure 3. Dynamic Determination of Executing Location

C. Data Mashup for Auditing Evidence

Furthermore, in order to guarantee that client data is processed appropriately in the client environment, both the communication logs with SaaS application and the file accessing logs in the client environment must be captured by Information Gateway. These logs must not be sent to the SaaS application in the cloud but must be stored in the Information Gateway. If some problem occurs in the client environment, the user must be able to accept these logs merged with cloud logs without influencing cloud service, such as auditing evidence, as illustrated in Figure 4.

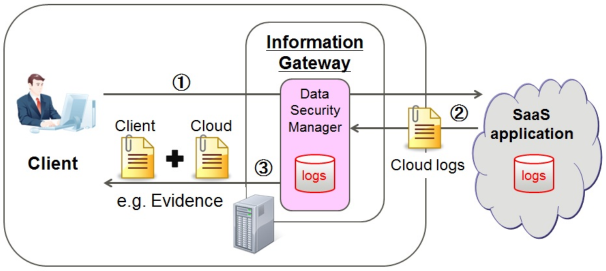


Figure 4. Data Mashup for Auditing Evidence

D. Portability for Existing Applications

It is an also important factor for existing applications to be easily regenerated into the SaaS application because there are some services that have already been provided. In order to reduce the workload of application developers, Information Gateway must offer complicated routing logic by simple description. If the routing logic can be expressed intuitively and comprehensibly, application developers have only to concentrate on describing the business logic.

III. Design of Framework

Figure 5 shows a design of our prototype framework. A SaaS application framework with Information Gateway is composed of integration framework, extended sandbox, and data security manager on the execution environment.

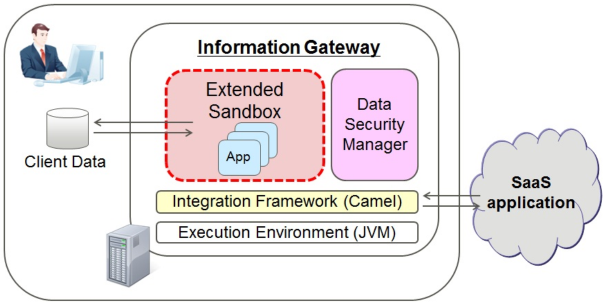
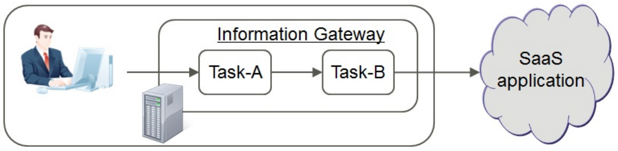


Figure 5. Components of Information Gateway

A. Integration Framework

The integration framework is core the component that enables the use of a SaaS application without sending confidential information to the cloud environment. In our prototype framework, we have adopted Apache Camel [6] [7] as the integration framework. The feature of Apache Camel is a routing and mediation engine. It allows application developers to define routing logic briefly, decide from which source to accept messages, and determine how to process and send those messages to other destination. By using a Domain-Specific Language (DSL), a chain of processing flow can be written in a simple manner.

An example of a routing logic defined using Java DSL is shown in Figure 6. In this example, after two tasks are processed in the client environment, only the processing result is routed to the SaaS application. To define routes of these tasks, only the “configure()” method of a “RouteBuilder” class has to be implemented. These routes begin with a “from” endpoint and finish at a “to” endpoint, respectively. All necessary business logic is implemented in between. In this example, since the case with existing applications, such as Task-A and B, is assumed, existing applications are wrapped and only set as an argument of “process()” method. It turns out that the routing logic can be expressed intuitively and comprehensibly using Java DSL.



```
public class JobRouteBuilder extends RouteBuilder {

   @Override
   Public void configure() {

      from(“direct:callTask-A”)
      .process(new Processor(Task-A)).to(“direct:callTask-B”)

      from(“direct:callTask-B”)
      .process(new Processor(Task-B)).to(“http://......../service”)

   }
}
```

Figure 6. Example of Routes with Java DSL

B. Extended Sandbox

Originally, the sandbox restricts access to local resources, such as client data and applications, from a third party’s application. In our framework, in order to capture the access to the local resources and the communication with SaaS application in the cloud, we have enhanced the functions of Java Virtual Machine (JVM). These captured logs are stored in Information Gateway, and are never sent to the cloud. By thus extending the functions of JVM, the client user is able to accept a value-added service, such as auditing evidence.

C. Data Security Manager

The data security manager is a component for judging where Information Gateway must execute processing based on both the data policy and the attribute of SaaS application. If a part of the processing of the SaaS application is executed in the client environment, the outsourcing vendor registers which processing must be executed in the client environment to the data security manager. Similarly, the client administrator (or user) registers a data policy indicating whether data contains confidential information or not. As a result, if the client user accesses the SaaS application in the cloud by way of Information Gateway, the Information Gateway can dynamically determine where the processing must be executed, or whether data must be encrypted or not.

IV. Case study

To verify that our framework was able to provide new service for the user securely and quickly, we have implemented a real consulting service in cooperation with outsourcing business divisions. We adopted an analysis service as a use case, which analyzes legacy codes and returns analysis results for the client user. Recently, the needs of service-analyzing source codes has increased more and more because legacy codes result in a complicated logic due to maintenance activities throughout long period of time.

Figure 7 shows a message flow of the implemented analysis service. Information Gateway integrates client data with analysis service in the cloud. Although source codes contain confidential information, all processing is executed securely by clicking an execution button. In this use case, if the client user accesses the analysis service in the cloud using Information Gateway, pre-processing and encrypting are executed in the client environment, and then the remaining processing is continued in the cloud. Therefore, our framework enables seamless analysis service without sending the confidential information to the cloud. Moreover, to provide auditing evidence service for the client user, the mashup function of Information Gateway also made it possible to merge client logs with cloud logs without influencing the analysis service. The client logs consist of file accessing logs and communication logs, which are captured with sandbox in the Information Gateway.

On the other hand, from the standpoint of application developers, we evaluated the workload required to add a new service on our framework. As shown in Table I, HTML and three Java Class files were needed. HTML was needed so that the client user may interface with the analysis service in the cloud. As for other Java Class files, the “RouteBuilder” class that defines the routing logic, the “Processor” class that defines the business logic, and the “Controller” class that controls the service were needed respectively. However, since there was an existing application in this case, SaaS application could be implemented only by rewriting the parameters of the default template class in our framework.

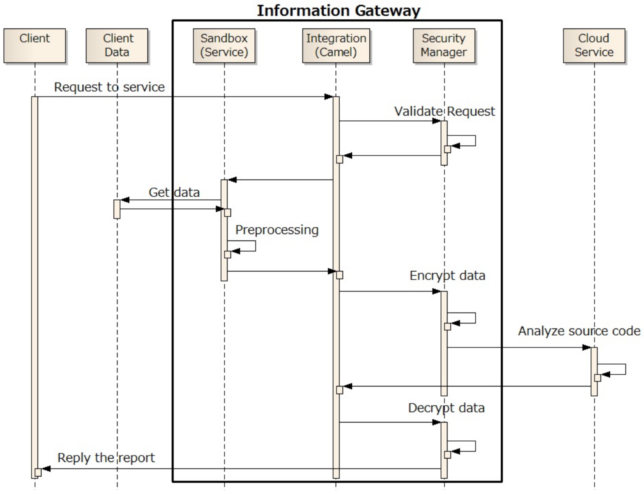


Figure 7. Message Flow of Analysis Service

Owing to our framework, a new analysis service could be implemented within an hour. In other words, the result of the case study shows that existing applications can be ported into SaaS application easily, and application developers have only to concentrate on business logic. And furthermore, we also confirmed that two or more services can be added, and these services work simultaneously, without interfering mutually.

Table I
EVALUATION RESULT OF WORK

HTML	Java Class	Work Time
1 file (4 lines per file)	3 files (3 lines per file)	<1.0 hour

V. RELATED WORK

A lot of proposals integrating client data and applications with cloud service have been performed so far. Feng Liu et al. [8] [9] proposed a system consisting of two types of proxy servers. One of the proxy servers is deployed in the cloud environment and the other in the client environment beforehand. If client user sends a request to the SaaS application in the cloud, the negotiation is done between these servers and a connection is established. Once the tunnel channel is established, the SaaS application can send the request to the client user. This means that the SaaS application in the cloud can acquire even client data containing confidential information without the permission of the user. And in that case, a configuration change on firewall is needed.

However, it is actually difficult to change the security rule in the organization. Especially, it becomes more important to handle client data carefully in such ICT consulting service. In our framework with Information Gateway, we propose the secure integration pattern with the SaaS applications without the configuration change on firewall, only by considering outbound requests from the client user.

VI. CONCLUSIONS

We present the design and implementation of a SaaS application framework using Information Gateway that enables the use of the SaaS application in the cloud while retaining data confidentiality. By setting up Information Gateway in the client environment, the executing location can be controlled dynamically based on the data policy and the attribute of SaaS application. As a result, the client user is able to use the SaaS application in the cloud without sending confidential information. Moreover, since the complicated routing logic can be defined briefly owing to our framework, existing applications can be ported easily into the SaaS application. Therefore, application developers have only to concentrate on business logic.

Currently, a trial experiment has already started using our framework in order to apply some real service. We evaluate the usability and the performance, and make further refinements to our framework.

REFERENCES

[1] S. Gagnon, V. Nabelsi, K. Passerini, and K. Cakici, “The next web apps architecture:challenges for saas vendors,” *IT Professional*, vol. 13 Issue:5, pp. 44–50, Oct. 2011.

[2] Q. Zhang, L. Cheng, and R. Boutaba, “Cloud computing:state-of-the-art and research challenges,” *Journal of Internet Services and Applications*, vol. 1, pp. 7–18, Jan. 2010.

[3] Z. Pervez, S. Lee, and Y.-K. Lee, “Multi-tenant, secure, load disseminated saas architecture,” in *Advanced Communication Technology (ICACT), 2010 The 12th International Conference*, Phoenix Park, Feb. 2010, pp. 214–219.

[4] S. Kang, S. Kang, and S. Hur, “A design of the conceptual architecture for a multitenant saas application,” in *Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference*, Jeju, Island, May 2011, pp. 462–467.

[5] S. Dhar, “From outsourcing to cloud computing:evolution of it services,” in *Technology Management Conference (ITMC), 2011 IEEE International*, San Jose, CA, Jun. 2011, pp. 434–438.

[6] Apache camel. [Online]. Available: http://camel.apache.org

[7] C. Ibsen and J. Anstey, *Camel in Action*, 1st ed., C. Kane, Ed. Manning, SC: Manning Publications Co., 2011.

[8] F. Liu, L. Li, and W. Chou, “Communications enablement of software-as-a-service (saas) applications,” in *Global Telecommunications Conference, 2009. GLOBECOM IEEE*, Honolulu, HI, Nov. 2009, pp. 1–8.

[9] F. Liu, W. Guo, Z. Q. Zhao, and W. Chou, “Saas integration for software cloud,” in *2010 IEEE 3rd International Conference on Cloud Computing*, Miami, FL, Jul. 2010, pp. 402–409.