SMEF: An Entropy-based Security Framework for Cloud-oriented Service Mashup

Ruixuan Li, Li Nie, Xiaopu Ma, Meng Dong, Wei Wang
Intelligent and Distributed Computing Lab, School of Computer Science and Technology
Huazhong University of Science and Technology, Wuhan 430074, P. R. China
rxli@hust.edu.cn, {linie, xpma, mengdong, weiwang}@smail.hust.edu.cn

***Abstract*—Cloud-oriented service mashup can aggregate many services to provide personalized services for end-users on demand. However, how to securely aggregate mashup services becomes a bottleneck of hampering the development of cloud computing. In this paper, we present a secure cloud service mashup framework called SMEF to address this problem. In SMEF, we employ security entropy to measure the unascertained security degree of service mashup. The nonfunctional criteria of SMEF are aggregated as a single criterion by defining a utility function. Then the relatively optimal mashup services are selected to meet the user requirements. Finally, we have implemented a simulation of SMEF and conducted extensive experiments using simulations of different sizes of services and security factors. Experimental results show the feasibility and efficiency of the SMEF service mashup framework.**

Keywords-cloud computing; security framework; service mashup; entropy theory

I. INTRODUCTION

As a new computing mode, cloud computing utilizes many traditional technologies, such as parallel computing, grid computing, distributed computing, network storage and virtualization. As a new business mode, it can integrate large-scale computation, storage, data and applications to provide various types of services [1], such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). In a cloud computing environment, users can access on-demand and pay-per-use services anywhere provided by the service providers. Now many enterprises have adopted this new mode to provide services because it has many advantages, such as convenience, economy, generality and high scalability.

In a cloud computing environment, service mashup [2] is a new technology that aggregates various services to form a new application for providing services according to user requirements. Compared with the traditional service composition, service mashup focuses more on users' requirements. It can use smaller lightweight services of more simple and easy implementation to create rich custom-built personalized services with efficiency, flexibility and reusability in cloud platform. Therefore, service mashup is gradually becoming a hotspot in cloud computing field.

Nowadays, more and more enterprises start to focus on service mashup. Some mashup platforms have been developed, such as FeedBurner by Google, QEDWiki by IBM, Pipes by Yahoo and Popfly by Microsoft. Furthermore, these famous IT companies also take mashup as an important approach for the next generation lightweight service composition, which assembles internet and enterprise content into simple, flexible, and dynamic web applications. In the research field, there are also more and more academic institutions to be attracted, and a few contributions [3, 4] have been made on this subject. However, most of these studies [5, 6] pay attention to the functional requirements and quality of service (QoS) of mashup, and little focuses on security aspect of service mashup.

As we know, virtualization technique that used in cloud computing environment enables the personal user data to be randomly distributed in different clouds, which poses various security and privacy challenges; such as security strategy confliction and malicious attack. Security is currently stated as an inhibitor for cloud adoption, which leads to around half of all IT managers in North America and Europe to decide against the use of cloud services. Hence, in order to enable enterprises and organizations to adopt cloud computing application platform on a large scale and securely providing services, we must comprehensively analyze and address security issues of service mashup in the cloud computing. Therefore, security issues in the cloud computing environment have become more and more important and challenging [7, 8].

In this paper, we present a secure service mashup framework, called SMEF, to address all of the above security problems. SMEF adopts a novel security service mashup assessment method to assess security risk of each service based on entropy theory [9], where the entropy can be used to thoroughly analyze the nature of service mashup security and comprehensively consider the effects of various safety factors. By computing service entropy and mashup services chain entropy, SMEF can choose a relatively optimal mashup service chain to meet both user functional and nonfunctional requirements, especially for user security requirements.

The main contributions of this paper are as follows:

- We present a secure framework for cloud-oriented service mashup. This framework can make sure that the mashup services simultaneously satisfy functional and nonfunctional requirements.
- We introduce the concept of entropy to assess the single service and mashup service chain in order to analyze the security of the cloud service mashup.

978-0-7695-4600-1/11 $26.00 © 2011 IEEE
DOI 10.1109/TrustCom.2011.41

- A multi-objective selection method for service mashup is proposed to look for mashup services, which meets the requirements of QoS and security.

The remainder of this paper is organized as follows. Section 2 discusses the related work. In Section 3, we introduce service entropy and service mashup chain entropy to quantify the security degree. In Section 4, a two-staged security framework is presented, functional requirement and non-functional requirement respectively. A summary of our experimental results on simulated data is discussed in Section 5. Finally, Section 6 concludes the paper and provides some insight into our ongoing and future work.

II. Related Work

Service Mashup first is clearly introduced by Djamal and Schahram [10] as a new generation of Web-based applications for achieving easy-to-accomplish end-user service compositions in 2008, to our knowledge. Recently, researchers have taken an active interest in this field. Most of researches focus on how to conceptualize, model and design the mashable services [11], how to construct generative environment for the orchestration of abstract services [12], and how to better satisfy functional and QoS requirements. However, there are few researches on the security requirements in service mashup.

In [13], Florian provides a model and semantics for integrating security concerns to address authentication and authorization from a language perspective. Jonas [14] presents a security lattice-based approach to mashup security by formalizing a notion of composite delimited release policy. However, they have no comprehensive consideration of security and the essence of security factors. In addition, these existing studies don't take into account the users' security needs and dynamic changes of cloud environment.

As the researches of service mashup are relatively rare, we need to borrow ideas from the existing researches of security service composition. In the field of the traditional service composition, some scholars have made beneficial researches and put forward some methods for security service composition. In [15], Sathiaseelan proposes a generic architecture called Multi-Level Secure Architecture (MLSA) exclusively for the academic institutions that provide integrated web services in a secured manner. In [16], Chen proposes a trust-sensitive strategy based on black and white plates, and the blackboard and whiteboard trust model could help users to get reliable, high-quality services composition.

From the security service mashup and service composition methods discussed above, it can be seen that the traditional method has many deficiencies such as the method requires a large number of extra overhead or only develops for a kind of specific security attacks. Furthermore, the service providers' characteristic such as distributive, self-adaptability, high dynamic and loose coupling also makes existing methods can not apply to the cloud service mashup. In addition, services mashup is dominated by end-users and has more flexibility. Hence, we need a new secure method to protect the security of cloud service mashup.

In this paper, we present a secure service mashup framework called SMEF to address security problems in service mashup. To the best of our knowledge, the utilization of entropy for analyzing secure cloud service mashup is novel in the area. The entropy can give a quantitative description of the uncertainty of security factors in cloud service mashup.

III. Multi-hierarchy Security Entropy Modeling

In cloud computing environment, the security of service mashup is affected by many security factors such as context environment, the user's security requirements and so on. Here, these security factors mutually interrelate and restraint, which makes the security factors to present multidimensional and multi-layered complexity. As the result of multi-criteria security features of the service mashup, the SMEF framework adopts Analytic Hierarchy Process to build security entropy model.

A. Establishment of Evaluation Index Set

In this paper, we firstly use the same method in [17] to divide the security factors of the cloud service mashup into user, service and environment. Thus, we can use these three major factors as the three factors to describe their influence on the cloud service mashup.

As shown in Fig. 1, the security factors can be described as a three- hierarchy structure:

- In the topmost layer, it reflects the environment factors impacted on the security, such as cyber attack, network communication, nature disaster caused by force majeure, et al.

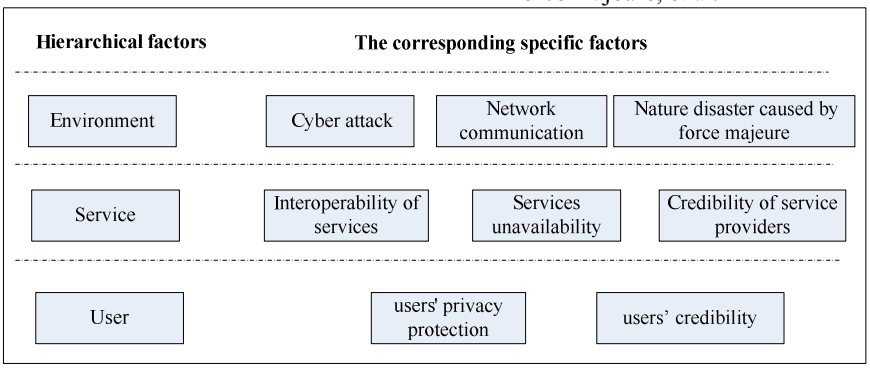

Figure 1. Three-hierarchy structure of security factors

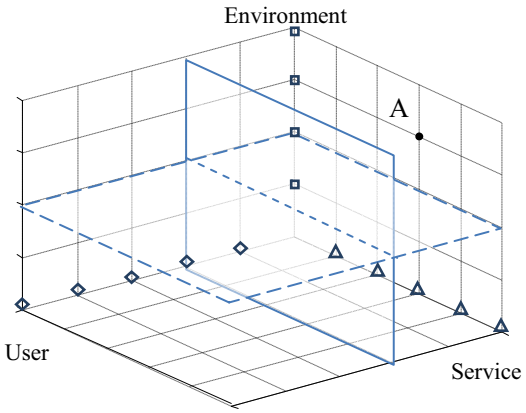

Figure 2. Three-dimensional model

- While in the middle layer is the service factor which can also affect the security of services mashup such as interoperability of services, services unavailability, credibility of service providers, et al.
- Finally, the user's security requirements consists of the lowermost layer, such as user privacy protection, user credibility, et al.

Among the three main factors, the topmost user security depends on the services security. In this situation, if service security is not guaranteed, the user cannot meet with the security needs. Similarly, the service security depends on the environment security.

The influences of these three main factors present three-dimensional characteristics. By synthetically considering inherent relations, user-environment-service multi-factor entropy model is constructed. For instance, in Fig. 2 the point A represents services and environment factors' impact on entropy value that under the condition where do not consider the factor of users.

Based on the hierarchy structure, the first level assessment factor set is constructed as (1).

$$U = \{u_1, ..., u_n\} \tag{1}$$

Where n=3, and it respectively represents user, environment and service factors. And each element in U is the second-level assessment factors set which is constructed as (2).

$$U_i = \{u_{i1}, u_{i2}, u_{i3}, ..., u_{im}\}, 1 \le i \le n \tag{2}$$

Where u_{ij} denotes the concrete security factor in the set U_i. For instance, if in formula (1) U_i denotes the environment factors, the u_{ij} represents the concrete security factor, which may be cyber attack in environment factor or other factors.

B. Security Quantification of Single Service

In our study, the security degree is defined as follows:

Definition 1 (*Security Degree*). Suppose there is a specified probability for the factor u_{ij} of the k_{th} services or a service to fulfill security needs. And this probability is denoted by r_{ij}^k, which quantifies possibilities of meeting corresponding security demand.

1) Constructing Security Degree Matrix

Every security factor is divided into *n* levels according to the capability of Security. The *n* levels in the security factor u_{ij} are denoted by $A_{ij} = \{a_1, a_2, \ldots, a_n\}$. For instance, for user privacy protection, the security factor can be divided into five levels: higher, high, average, low, lower. Furthermore, different security levels have different effects on the security, so the coefficient c_k is introduced to denote the influence for security of the level a_k, which enables the security factor to uniform the capability of security protection at all levels. The corresponding set is denoted by $C_{ij} = \{c_1, c_2, \ldots, c_n\}$. In addition, p_i represents the probability of the security factor is at the a_i level. $P_{ij}^k = \{p_1, p_2, \ldots, p_n\}$ represents the probability set for a security factor in the concrete k_{th} service, which can be quantified by context awareness function and historical information. The probability distribution of security capability on a security factor u_{ij} of the k_{th} service can be expressed as follows:

$$\left(A_{ij}^k, C_{ij}^k, P_{ij}^k\right)^T = \begin{pmatrix} a_1 a_2, ..., a_n \\ p_1 p_2, ..., p_n \\ c_1 c_2, ..., c_n \end{pmatrix}_{ij}^k \tag{3}$$

Where $\sum_{i=1}^{n} p_i = 1$. According to the above formula, the entropy of the security factor u_{ij} in the k_{th} service can be calculated by the assessment set.

$$h_{ij}^k = -\sum_{i=1}^{n} c_i p_i \log p_i \tag{4}$$

Here, if all c_i in the set c_{ij} are equals, it can be inferred that:

- If the values of all p_i are equal, all levels of the degree needn't distinguishing. In this case, the corresponding entropy reaches its maximum, which meets the maximum discrete entropy theorem.
- If $p_i = 1$, and the others N-1 are 0, the corresponding entropy reaches its minimum.

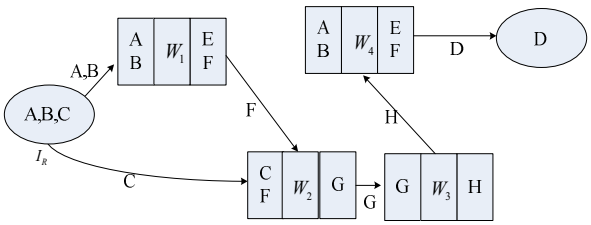

Figure 3. An example of mashup service chain

There has been some study with respect to the relations of the entropy and the security uncertainties [18]. In particular, the following equality holds for the inherent link between the entropy and the security degree.

$$r_{ij}^{k} = c_{ij} e^{-h_{ij}^{k}} \tag{5}$$

Where r_{ij}^{k} is the security degree of the security factor u_{ij} in the k_{th} service, and h_{ij}^{k} is a corresponding entropy value. c_{ij} is the correspond coefficient, which enables $r_{ij}^{k} \in [0,1]$ .The value of r_{ij}^{k} can be quantified according to formula (5). Furthermore, security degrees of every factors for the k_{th} service are denoted as follows.

$$(R_{ijk})_{i\times j} = \begin{bmatrix} r_{11k} & r_{12k} & \cdots & r_{1mk} \\ r_{11k} & r_{11k} & \cdots & r_{11k} \\ \vdots & \vdots & \ddots & \vdots \\ r_{11k} & r_{11k} & \cdots & r_{nmk} \end{bmatrix} \tag{6}$$

Where there is $i \in \{1,2,\ldots,n\}$, $j \in \{1,2,\ldots,m\}$ and $r_{ij}^{k} \in [0,1]$.

2) Determining Entropy Weight of Every Factors

Based on the security degree of all services, we can calculate the proportion of each security factor u_{ij} of the k_{th} service in security factors u_{ij} , which is denoted by (7).

$$f_{ij}^{k} = r_{ijk} / \sum_{t=1}^{s} r_{ijt} \tag{7}$$

Where s denotes the number of all services. Hence, we can get the entropy of the evaluated security factor u_{ij} and the entropy weight of every security factors as follows.

$$w_{ij} = -\sum_{k=1}^{s} f_{ij}^{k} \log f_{ij}^{k} \tag{8}$$

$$s_{ij} = \frac{1 - w_{ij}}{m \times n - \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij}} \tag{9}$$

The value of s_{ij} can be quantified according to formula (9). Furthermore, The entropy weight of every factors can be expressed by (10).

$$(S)_{i\times j} = \begin{bmatrix} s_{11k} & s_{12k} & \cdots & s_{1mk} \\ s_{21k} & s_{22k} & \cdots & s_{2mk} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n1k} & s_{n2k} & \cdots & s_{nmk} \end{bmatrix} \tag{10}$$

Where there is $i \in \{1,2,\ldots,n\}$ and $j \in \{1,2,\ldots,m\}$.

3) Calculation of Service Entropy

According to (6) and (10), we can get the evaluated security service entropy integrating single-factor entropy as follows.

$$H(K) = -\sum_{i=1}^{n} \sum_{j=1}^{m} s_{ijk} \times r_{ijk} \log r_{ijk} \tag{11}$$

4) Security Quantification of Mashup Service Chain

Definition 2 *(Mashup Service Chain)*. A mashup service chain is a collection of services and their relationship for meeting some specific demand. An interface parameter set (s_{in}, s_{out}) is defined for each service, which leads to the sequence of related services.

Fig. 3 shows a typical mashup service chain, which forms a set of mashup service scheme. $F(A,B,C,D,E,F,G)$ is an interface parameter set. Due to interfaces restrictions, there is the corresponding order of implementation of the services set $S(w_1, w_2, w_3, w_4)$. For instance, the service w_3 must be executed before the service w_4 .

5) Calculation of Conditional Entropy and Joint Entropy

Taking into account the interactions of the mashup services, we apply the concepts of mutual information and conditional entropy in information theory to define different types of service mashups. We use X and Y to express two different services, and the Joint Entropy $H(X,Y)$ expresses X and Y service's security uncertainty :

$$H(X,Y) = -\sum_{i=1}^{n} \sum_{j=1}^{m} r(u_{ij}^{x}, u_{ij}^{y}) \log r(u_{ij}^{x}, u_{ij}^{y}) \tag{12}$$

The conditional entropy *H(Y/X)* expresses the *Y* service's security uncertainty in the case of the security of service *X*:

$$H(Y/X) = -\sum_{i=1}^{n} \sum_{j=1}^{m} r(u_{ij}^{x}/u_{ij}^{y}) \log r(u_{ij}^{x}/u_{ij}^{y}) \tag{13}$$

When the X and Y are independent, the formula can be recorded as：

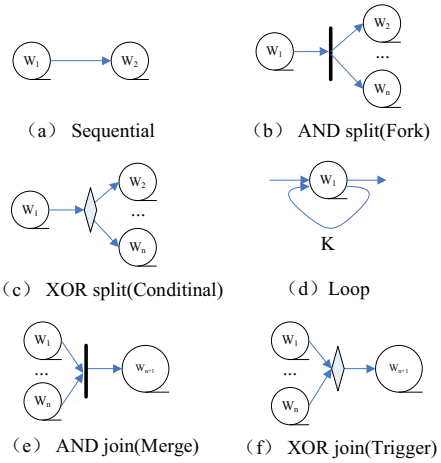

Figure 4. Mashup structure patterns

$$H(Y/X) = H(Y) \tag{14}$$

The relationship of the conditional entropy is:

$$H(X,Y) = H(X) + H(Y/X) = H(Y) + H(X/Y) \tag{15}$$

6) Calculation of Mashup Service Entropy

Atomic services may be connected by different structures in a composite service. Fig. 4 shows the six service composition structures considered in our study [19]: Sequential, AND split (fork), XOR split (conditional), Loop, AND join (Merge) and XOR join (Trigger). We can divide these six patterns into three categories: one-to-one mapping and many-to-one mapping. For example, in the AND split, w_1 can trigger w_2 and w_3, and this pattern can be split into two one-to-one mapping: $w_1 \to w_2$ and $w_1 \to w_3$; in the And Join, all services during w_1 and w_n are needed in order to trigger w_n, and this pattern belongs to the many-to-one mapping.

- *One-to-one mapping*

In this situation, conditional entropy can be adopted to solve this category. Suppose this one-to-one mapping is X and Y, so the entropy of this category can be defined as :

$$H(X \to Y) = H(X) + H(Y/X) \tag{16}$$

- *many-to-one mapping*

In this situation, the combination of conditional entropy and joint entropy can solve this category. Suppose this many-to-one mapping is $(X_1,\ldots,X_n)$ and Y, so the entropy of this category can be defined as :

$$H((X_1,\ldots,X_n) \to Y) = H(Y/X_1,\ldots,X_n) + H(X_1,\ldots,X_n) \tag{17}$$

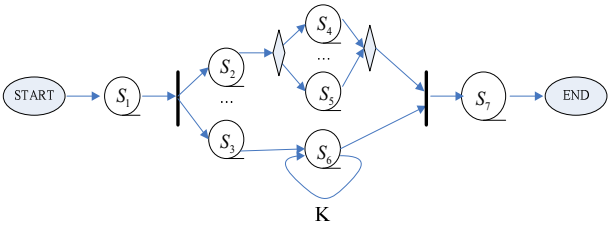

Figure 5. An example mashup services chain

- *One-to-many mapping*

In this situation, the combination of conditional entropy and joint entropy can solve this category. Suppose this one-to-many mapping is Y and $(X_1,\ldots,X_n)$, so the entropy of this category can be defined as :

$$H(Y \to (X_1,\ldots,X_n)) = \max(H(Y \to X_1), H(Y \to X_2),\ldots H(Y \to X_n)) \tag{18}$$

7) Calculation of Mashup Services Chain Entropy

Suppose that a mashup service is composed of a mashup service chain $S(s_1,s_2,\ldots,s_n)$.

As shown in Fig. 5, the mashup service chain [19] is made up of the above structure patterns. And the entropy of mashup services chain S can be expressed by three categories described above.

IV. SMEF Framework for Cloud Service Mashup

The SMEF framework can be decomposed into two stages. The first stage deals with the desired functionality requirements of the mashup services which called the functional mashup. The second stage involves QoS, constraints of security requirements based on the entropies of single service and mashup service chain which called the nonfunctional mashup. In the first stage, the SMEF supports a variety of ways to ensure functional satisfaction guarantee. In the second stage, the security is measured by the entropy values, and then the SMEF chooses a multi-objective mashup service schemes from both users' QoS and security. Here, the second stage about security is our focus.

A. SMEF Architecture

As shown in Fig. 6, SMEF consists of four components: functional mashup, service entropy quantification, mashup chain entropy quantification and nonfunctional mashup. We describe the details below.

- *Functional mashup:* giving out formalization description to generate an abstract service interface, then the AI planning-based algorithm for service composition is adopted to satisfy user functional requirement.
- *Service Entropy Quantification:* in this module, the security factors is extracted and classified, then use AHP to quantify security degree of each security factor. Finally, single-factor security degree is integrated to quantify security entropy of each service.

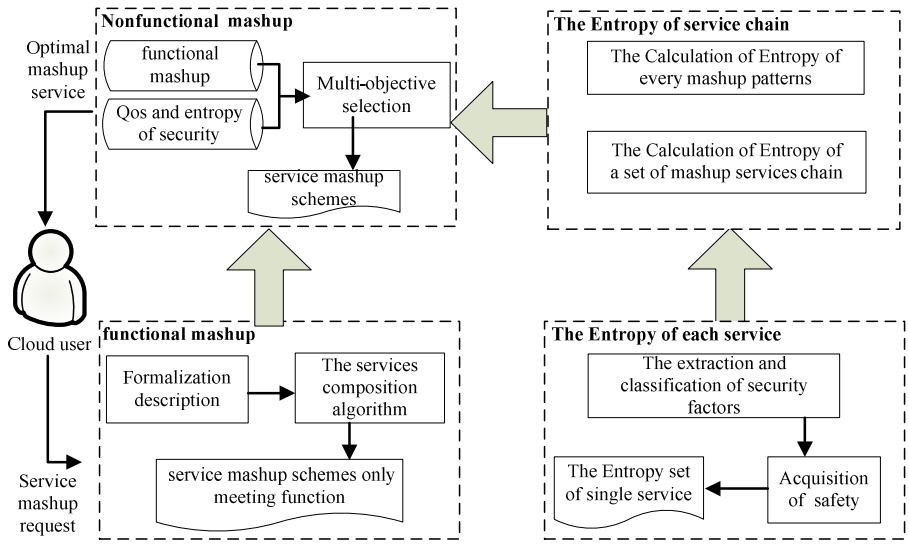

Figure 6. Simplified architecture of SMEF

- *Mashup chain Entropy Quantification:* in this module, six service structure patterns are divided into three categories (One-to-one mapping, One-to-many mapping, Many-to-one mapping). And conditional entropy and joint entropy are combined to solve these three categories. Finally, a service chain is split according to these three categories and quantifies the mashup chain.
- *Nonfunctional mashup:* this module is to ensure non-functional constraints including QoS and security, which can be used as an issue of service selection with multi-objective optimization.

In this paper, we focus on the nonfunctional service mashup. And the multi-objective mashup selection techniques of Fig. 6 are explained in the next.

B. Selection of Nonfunctional Mashup

Since there are many nonfunctional criteria for a web service, it is difficult to state that one service chain is better than another unless we compare them under a single criterion. The usual approach is defining a utility function, as a single criterion, that aggregates multiple criteria. In this paper, we consider the common case that users want to minimize the entropy and time while maximize the reliability and throughput at the same time. We therefore define the utility function as a weighted aggregate of these four factors:

$$F(k) = (w_1 Q_e(k) + w_2 Q_t(k) / w_3 Q_r(k) + w_4 Q_h(k)) \qquad (19)$$

Where w_1, w_2, w_3, w_4 are pre-defined weights of entropy, time, reliability and throughput respectively. $Q_e(k)$ is calculated by formula (10), and $Q_t(k)$, $Q_r(k)$ and $Q_h(k)$ are functions that normalize the generic QoS criteria between 0 and 1. Users can create constraints that capture their certain requirements for quality criteria. We here give four typical constraints: (1) $Q_e(k) \geq q_e$; (2) $Q_t(k) \geq q_t$; (3) $Q_r(k) \geq q_r$; (4) $Q_h(k) \geq q_h$. Where q_e , q_t , q_r , q_h are given thresholds. More complicated constraints on quality criteria can be produced by combining these typical constraints.

Algorithm 1 Service Mashup for Non-functional request including security

```
Algorithm Nonfunctional-Satisfy(S)
Input Service chain(S)
Output: True, false: S is satisfied or not
1: Rewrite criterion to normal form
2: For each (S)
3:    For each (k ∈ S)
4:            If (F(k) ∉ range)then
5:          Return false;
6:        End if
7:    End for
8:    If (Qe(k) ≥ qe)
9:        Return false;
10:   End if
11: End for
12: Return true
13: End
```

Through using Algorithm 1 we can find the desired mashup service chain satisfying the nonfunctional requirements from the all mashup service chains satisfying the nonfunctional requirements.

V. Experiment and Evaluation

We generate a large number of random services and mashup requests in order to analyze the efficiency of our

TABLE I. MEANINGS OF EVERY PARAMETERS IN THE SIMULATION PROGRAM

Parameters	Meanings
NA	The scale of atomic Services
NS	The number of security factors of each service
AdjSd	As variable parameter, security degrees of preset services are adjusted dynamically

framework in order to analyze the efficiency of our framework.

A. Experiment Preparation

Since it is difficult to construct a service mashup platform in real cloud computing environments, we carried out the experiments based on simulation programs. The base line is the system presented in China Web Service Cup (CWSC2011). We have implemented the experiments on a 2.10GHZ Intel Pentium 4 PC with 1GB main memory. The running platform is Microsoft Windows XP. All the algorithms are written in Java. The parameters and their meanings are listed in Table I.

B. Experiment data and analysis

Making use of these three adjustable parameters in table 1, we conducted three groups of comparative tests. We preset security degrees of some services and make the values relatively low. In theory, the lower the values of security degrees of the service, the lower the security of corresponding services. These services with low security degrees should avoid to be chosen. Hence, the selection ratio of this kind of services is be defined as FSR(false selection rate) to evaluate the system security performance. At initialization, randomly positive integers are generated to simulate the security degree of the service in the simulation program. In order to make user of the FSR, we assume that the security degrees of a third of services fit a normal distribution with mean 0.2, while security degrees of the rest service obey the normal distribution with mean 0.5.

We present the evaluation of our algorithm on the different parameters. To get more accurate experimental results, we have run each experiment for 100 times and take the average as the result. The front two parameters in Table I are selected to set 2000 and 3 respectively as the reference value. And these parameters are adjusted to analyze the changing regularity of mashup effect. 5 indexes are designed to validate the SMEF framework, and they are: TNS: cost time of finishing a mashup service not considering security; TS: cost time of finishing a mashup service considering security by using of SMEF; SNS: average success rates for mashup requests not considering security; SS: average success rates for mashup requests considering security by using of SMEF; FSR: the percentage of chosen services from all the preset services of low security degrees.

Table II shows the values of corresponding indexes would bring about corresponding change when only the number of atoms services is changed and the other factors remain unchanged. By the analysis of the data above it can be found that with the number of atomic services increases, the cost time of SMEF has bearable increase compared with the situation without considering security. We can also see that the average completion rate for mashup requests also increased. In addition, FRS takes on slowly rising trend, which means that when the number of atoms service increases, the security of the system based on security entropy still can be guaranteed.

TABLE II. THE EFFECT FOR SERVICE MASHUPS INFLUENCED BY NA

NA	TNS (ms)	TS (ms)	SNS (%)	SS (%)	FRS (%)
2000	189.14	220.91	90	37	3.76
4000	363.67	404.73	93	34	4.07
6000	332.35	376.44	96	48	5.30
8000	634.23	672.12	97	56	4.98
10000	632.97	674.32	95	54	6.21
12000	823.32	875.23	96	56	6.32

Table III shows that the values of corresponding indexes would bring about certain changes when the number of security factors is changed and the other parameters aren't changed. By the analysis of the data above it can be found that when the number of security factors increases and security factors are considered more and more comprehensively, cost time of finishing a mashup service increases and the values of the FSR has a slowly downward trend. Furthermore, when the number of security factors is between 18 and 24, the values of SS is nearly unchanged, and the FSR is the lowest.

In the last set of experiment, we investigate frequency that one service will be selected by security service chains, when its security degree decreases or increases sharply. By the change of security degree it can be found that if a service increases its security degrees, the selected probability of this service will increase correspondingly. This result is consistent with not only the starting point of the entropy modeling, but also actual needs.

TABLE III. THE EFFECT FOR SERVICE MASHUPS INFLUENCED BY NA

NS	TNS (ms)	TS (ms)	SNS (%)	SS (%)	FRS (%)
3	232.14	300.15	92	46	9.03
6	232.14	333.32	92	43	10.07
9	232.14	341.42	92	34	5.30
12	232.14	321.53	92	32	8.98
15	232.14	348.29	92	38	6.21
18	232.14	339.53	92	37	6.32
21	232.14	342.63	92	35	7.21
24	232.14	347.34	92	31	5.32

By the experiment and data analysis described above, we can conclude that system uptime has acceptable increases when we use the SMEF framework to consider security factors based on Entropy. In addition, when selected parameter is located in suitable range, this SMEF framework can achieve better security effect of service mashup. Furthermore this experiment only considers the general service mashups, characteristics of entropy can better meet with practical applications under the cloud environment. Thus, in this paper, the framework based on entropy of security service mashup, can achieve the desired effect.

VI. CONCLUSIONS

In this paper, we presented a secure framework SMEF for service mashups in cloud computing environment. The core security mechanism utilizes the entropy to assess the security of service mashups in consider security factors which can be decomposed into two stages: functional mashup and nonfunctional mashup. In this framework, it firstly ensures the services' functional requirements, and then chooses a multi-objective security mashup service schemes to meet user requirements of QoS and security. Finally, it can choose a relatively optimal mashup service chain meeting both user functional and nonfunctional requirements. There are several directions for future work to further improve our framework. One thread in our future work will focus on the utilization of the entropy to monitor and resolve strategy conflict because there are different strategies from different clouds.

VII. ACKNOWLEDGEMENTS

This work is supported by National Natural Science Foundation of China under Grants 61173170, 60873225, National High Technology Research and Development Program of China under Grant 2007AA01Z403, Natural Science Foundation of Hubei Province under Grant 2009CDB298, Wuhan Youth Science and Technology Chenguang Program under Grant 200950431171, and Innovation Fund of Huazhong University of Science and Technology under Grants 2011TS135 and 2010MS068.

REFERENCES

[1] R. Buyya, "Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities," in Proc. 10th IEEE International Conference on High Computing and Communications, Shanghai, Sept. 2008, pp. 5-13, doi:10.1109/HPCC.2008.172.

[2] D. Benslimane, S. Dustdar and A. Sheth, "Services Mashups: The New Generation of Web Applications," IEEE Internet Computing , vol.12, no.5 , Oct. 2008, pp.13-15, doi:10.1109/MIC.2008.110.

[3] Huiyang Xu, Meina Song, and X. Luo, "A QoS-oriented Management Framework for Reconfigurable Mobile Mashup Services," in Proc. 11th International Conference on Advanced Communication Technology, Gangwon-Do, South Korea, Feb. 2009, pp.15-18.

[4] Chan Yew Hiok, Vincent K.T Khoo, "Education Services Mashup: Examining the Impact of Web Design Features on User Trust," in Proc. Second International Conference on Computer Research and Development (ICCRD 2010), Kuala Lumpur, Malaysia, May 2010, pp. 349-353, doi:10.1109/ICCRD.2010.28.

[5] Gerardo Canfora, Massimiliano Di Penta, Raffaele Esposito, and Maria Luisa Villani, "QoS-Aware Replanning of Composite Web Services," in Proc. IEEE International Conference on Web Services (ICWS), Orlando, July 2005, pp. 311-327, doi:10.1109/ ICWS.2005.96.

[6] Liangzhao Zeng, Boualem Benatallah, Marlon Dumas, Jayant Kalagnanam, and Henry Chang, "QoS-Aware Middleware for Web Services Composition," IEEE transactions on software engineering, vol.30, no.5, May 2004 , pp.311-327, doi:10.1109/TSE.2004.11.

[7] H .Takabi, J.B.D. Joshi, and G. Ahn, "Security and Privacy Challenges in Cloud Computing Environments," IEEE Security & Privacy, vol.8, no.6, Nov.2010, pp.24-31, doi: 10.1109/MSP.2010. 186.

[8] H. Takabi, J.B.D. Joshi , A. Gail-Joon, "SecureCloud: Towards a Comprehensive Security Framework for Cloud Computing Environments," in Proc. IEEE 34th Annual Computer Software and Applications Conference Workshops, July 2010,pp. 393-398, Seoul , Korea, doi:10.1109/COMPSACW.2010.74.

[9] D.K .Kondepudi, Prigogine, Modern thermodynamics: From heat engines to dissipative structures, John Wiley & Sons,1998.

[10] D .Benslimane, S. Dustdar, and A. Sheth, "Services mashups: The new generation of web applications, " IEEE Internet Computing, vol.12, no.5, Oct. 2008, pp.13-15, doi:10.1109/MIC.2008.110.

[11] C. Mao, "Towards a more effective mashup using mashable service model, "Proceedings of the 2010 IEEE Congress on Services (SERVICES-1) , July 2010, pp. 566-573, IEEE Computer Society, Miami, doi:10.1109/SERVICES.2010.30.

[12] 1S.Chollet, P. Lalanda, "An extensible abstract service orchestration framework ," in Proc. 2009 IEEE International Conference on Web Services, ICWS 2009, July 2010,pp. 831-838, Los Angeles, CA, United states, doi:10.1109/ICWS.2009.14.

[13] F.Rosenberg, et al, "End-to-end security for enterprise mashups , Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),Service-Oriented Computing," in Proc. 7th International Joint Conference,ICSOC-ServiceWave 2009, pp. 389-403, Springer Verlag, Stockholm, doi:10.1007/978-3-642-10383-4_28.

[14] J.Magazinius, A.Askarov, and A .Sabelfeld, "A lattice-based approach to mashup security," in Proc. 5th International Symposium on Information, Computer and Communications Security,ASIACCS 2010, April.2010, pp.15-23, Association for Computing Machinery, Beijing, doi:10.1145/1755688.1755691.

[15] J.G.R Sathiaseelan, S.A. Rabara , and J.R. Martin, "Multi-Level Secure Architecture for distributed integrated Web services, " in Proc. the 2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT 2010) , July 2010,pp. 180-184, Chengdu, doi:10.1109/ICCSIT.2010.5563785.

[16] Z.G.Chen, L.P. Liu, A.F. Liu, "Trust-sensitive Web service composition strategy based on black and white board, "Tongxin Xuebao/Journal on Communications, vol.31, no.6, June 2010, pp. 25-34.

[17] JingLin Zhang, Safety systems engineering, Coal Industry Press, Beijing ,2002.

[18] G.Xi, et al, "The risk assessment of crime prevention system based on risk entropy model," in Proc. 2nd International Conference on Computer Research and Development, ICCRD 2010, Kuala Lumpur, Malaysia, Sept. 2010, pp.681-685, doi:10.1109/WICOM. 2010.5601335.

[19] T.Yu, Y. Zhang , K.J. Lin, " Efficient algorithms for Web services selection with end-to-end QoS constraints," ACM Transactions on the Web, vol.1, no.1, May 2007. doi:10.1145/1232722.1232728.