A COMPETITIVE PENALTY MODEL FOR AVAILABILITY BASED CLOUD SLA

Xiaoyong Yuan2, Hongyan Tang1, Ying Li*, Tong Jia1, Tiancheng Liu3, Zhonghai Wu1

1 School of Software and Microelectronics, Peking University, Beijing, China
2 National Engineering Center of Software Engineering, Peking University, Beijing, China
3 Computer & Information Science & Engineering Department, University of Florida, Florida, USA

Abstract

Availability is one of the most essential attributes of qualities of cloud services. Most popular public cloud services claim availability commitments with corresponding penalties in their SLAs. To gain the maximal profits, cloud providers should choose an optimal penalty strategy in the competitive cloud market. In this paper, we firstly survey availability and penalty calculation methods of cloud providers. Based on the survey, we propose a competitive penalty model and a corresponding penalty based profit maximization algorithm for cloud providers. According to the model, each cloud provider would choose the best fit penalty strategy to gain the maximal expected profit during the game procedure. The proposed model is evaluated with real data of popular cloud providers with sensitive analysis, and is valuable for cloud providers to define their penalty strategy.

Keywords: cloud service, SLA, availability, penalty degree

1. INTRODUCTION

Cloud computing utilizes internet to accomplish convenience and pay-as-you-go access to the shared resources in the process of provisioning and releasing resources (Mell and Grance, 2011). With this business model, it can largely save money for companies and organizations by taking place of traditional datacenter (Armbrust et al., 2010). Service Level Agreement (SLA) of cloud service is essential for cloud providers to guarantee their quality of cloud services for cloud consumers and maintain a good business relationship between cloud providers and cloud consumers. As an important part of cloud SLA, availability commitment is what cloud consumers concern about most. According to the survey (Cloud or Fog, 2009), 73% of CIOs and CFOs are reluctant to put their financial and accounting applications in the cloud and 57% of them would not place any business critical applications in the cloud due to their concerns about the potential downtime of the cloud. A recent report on cloud service outage (CloudHarmony; Butler, 2015) shows that Amazon EC2 underwent 20 outrages with total downtime of 2.41 hours in the last year; Microsoft Azure underwent 92 outrages and the total downtime summed up to 10.97 hours; Google GCE underwent 72 outrages and its downtime was 4.42 hours.

Cloud providers compensate the economic loss of cloud consumers due to potential downtime by defining their penalty strategies in their SLAs (Qiu et al., 2013). For example, Amazon EC2 claims to pay back 10% service credit to cloud consumers if its availability rate in one month is less than 99.95% but equals to or greater than 99.0%. If its availability rate is less than 99.0%, Amazon EC2 will pay 30% service credit. We investigate SLAs of 14 well-known public cloud services including Aliyun Cloud, Amazon EC2, Windows Azure, Tencent Cloud, Ucloud, Telecom Cloud, JDHosting, Baidu Cloud Compute, 360Kaopu Cloud, Rackspace, Google GCE, IBM Softlayer, VPS.net, and GoGrid. Our survey shows the penalty strategies among cloud providers are totally different. Some providers won’t pay much for even a long period of downtime, while others pay almost as much as their income for penalty. An extreme example is JDHosting (a public cloud provider in China) even pays twice as its income if its availability commitment is not satisfied, which is considered as a punitive penalty. The variability of availability commitment and penalty implies a corresponding lack of clarity in cloud SLAs, which prompts us to systematically analyze penalty strategy in cloud SLA.

In this paper, we firstly survey and analyze various availability and penalty calculation methods adapted by well-known cloud providers. Then we present a competitive penalty model for availability commitment in cloud SLA. The model aims to provide guidance for the cloud providers to maximize their profits by defining a suitable penalty strategy in cloud SLA. In this model, we assume that cloud...
consumers prefer to choose and buy the best offer from cloud providers, and cloud providers intend to attract more cloud consumers and thus earn more in the cloud market. Cloud consumers’ choices are based on their profits under the conditions of different availability rate, service price and penalty strategy among cloud providers. However, the choices are not the final decision as cloud providers would adjust their strategies and cloud consumers should choose again for the sake of profits. This process can be viewed as a game procedure where both cloud providers and cloud consumers are eager to maximize their profits. We introduce a stochastic game to simulate this process. In this paper, we propose penalty degree (Section 4) as a fixed ratio of downtime to describe the amount of cloud provider’s penalty and indicate their penalty strategy.

To gain the optimal penalty degree for each cloud provider, we design a penalty based profit maximization algorithm, which is consisted of the game procedure and choosing penalty strategies for cloud providers. The algorithm is verified with real data from cloud providers. A sensitive analysis is also conducted for this algorithm. Though our focus is IaaS providers, we believe the competitive penalty model can be a reference for PaaS or SaaS providers. To our best knowledge, we are the first to explore and provide guidance for penalty strategies.

Our contributions in this paper include: 1) an analysis of availability and penalty calculation methods of 14 well-known cloud services; 2) a competitive penalty model representing the game procedure between cloud providers and cloud consumers; 3) a Penalty Based Profit Maximization Algorithm to maximize the profit for cloud providers; 4) the evaluation of proposed model and algorithm.

The rest of paper is organized as follows. Section 2 discusses the related work. Section 3 elaborates an analysis on well-known cloud providers including various availability and penalty calculation methods. Section 4 proposes a competitive penalty model based on cloud SLA presented in Section 3 as well. Based on the penalty model, we describe the game procedure and design a Penalty Based Profit Maximization Algorithm. In Section 5, we evaluate the algorithm with real data and conduct a sensitive analysis. Finally, Section 6 concludes our work.

2. Related Work

2.1 Study on Cloud SLA

Patel et al. propose an architecture for managing cloud SLAs based on Web Service Level Agreement (WSLA) specification (Patel et al., 2009). In their work, WSLA is applied to describe the cloud SLA. They present three common WSLA services to achieve the automation of cloud SLAs based on WSLA. Their work also includes trusted third parties to take the task for better securities. Serrano et al. propose a novel cloud model called SLAaaS (SLA-aware-Service) in order to apply SLA into the cloud (Serrano et al., 2013). A specific language is introduced to describe QoS-oriented SLA associated with cloud services, dealing with QoS uncertainty and cloud fluctuations. Wang et al. propose a platform of SLA management where providers publish their SLA and consumers negotiate and choose the best satisfied providers (Wang et al., 2014). Moreover, there is a reputation system to evaluate the reliability of provider in the platform.

Alhamad et al. present main criteria, which should be considered in the SLA for IaaS, PaaS and SaaS. For example, in IaaS platform SLA metrics include: CPU capacity, memory size, boot time, storage, scale-up time, scale-down time, auto scaling, response time and availability (Alhamad et al., 2010). Availability is the feature guaranteed by all SLAs in their examination. However, their analysis about availability commitment is lack of details and irrelevant to the penalty. They compare SLA among Amazon EC2, Microsoft Azure Storage, Rackspace Cloud Servers, Dell Boomi and Google Cloud Storage in the aspects of claimed service availability, SLA outlining, establishing, penalty and exclusion. However, their work aims to all kinds of cloud services (IaaS, PaaS, and SaaS), and hasn’t specific views for IaaS like our paper.

Sfondrini et al. survey 58 IT companies who have deployed their applications in the public cloud (Sfondrini et al., 2015). They find that SLAs of cloud providers are lack of clear definition, especially in response time, data security and performance monitoring. Moreover, after exploring SLA metrics and comparing SLAs of current major cloud service providers, Aljoumah et al. reveal that definition of SLA lacks standard, particularly in monitoring (Aljoumah et al., 2015). Baset investigates SLAs of compute and storage services offered by 5 IaaS and PaaS providers (Baset, 2012). A cloud SLA is proposed to break down into several components and then different cloud providers can be compared based on these components. Alkandari and Paige did the same work to decompose SLA and propose a metamodel to define cloud SLA for both cloud providers and cloud consumers (Alkandari and Paige, 2012). Thus, consumers can compare cloud services and their QoS values. Based on Alhamad and Baset’s work, Qiu et al. analyze 29 SLAs of current public cloud services, including 17 SLAs for IaaS (Qiu et al., 2013). They identify a few commonly stated attributes and missing attributes which are important to both the providers and consumers. They find that attributes closely related to customer data, like security, privacy, protection and backup policies, are missing in most SLAs.

Due to growing number of cloud service offering, Redl et al. propose an automatic method to match SLAs between cloud providers and cloud consumers by means of machine learning algorithms (Redl et al., 2012). According to SLA metrics, parameters mentioned above, they are able to create SLA templates and automatically bind SLA elements in the templates with users’ history requirements.
2.2 Game Theory Based Cloud SLA Model

Game theory is one of the techniques applied in the bargaining-based negotiation where cloud provider and cloud consumer will make offer and counteroffer until they reach an agreement (Hani et al., 2015). To capture the SLA negotiation process, Figueroa et al. develop a bargain model based on game theory (Figueroa et al., 2008). In their model, consumers are offered an opportunity to express their preferences via a counteroffer instead of a take-it-or-leave-it contract. Provider and consumer bargain iteratively to maximize their payoff function until a perfect Bayesian equilibrium is achieved. Almeida et al. present a joint admission control and resource allocation technique to maximize Web service providers’ profits in (Almeida et al., 2010). If response time of a single Web service becomes longer than guaranteed response time in SLA (SLA violation), the providers will pay their revenue as penalty to customers. They also take into account the providers’ cost which is proportional to the use of physical capacity and time interval. Silaghi et al. propose a general framework of SLA negotiation strategy under time constrained in competitive computational grids and build it based on Bayesian learning agent (Silaghi et al., 2012). Their experiments show that opponent learning-based negotiation strategy can achieve optimal resource allocation and fair satisfaction of participants. Ray et al. propose a game theory based automatic SLA negotiation model, where a cloud service broker helps to achieve optimal SLA values of price and quality for both providers and consumers (Ray et al., 2014). Wang et al. consider SLA-based resource allocation problem among multiple cloud service providers in the cloud computing market as a normal-form game (Wang et al., 2014). They prove the existence of Nash Equilibrium and find the optimal strategy that maximizes profit for each provider.

Truong-Huu et al. propose a dynamic cloud resource-pricing model (Truong-Huu et al., 2014). They consider a competitive cloud market consisting of multiple providers and consumers. The competition among providers is formulated as a non-cooperative stochastic game and modeled as a Markov Decision Process. A discrete choice model is employed to derive consumers’ choice probability, according to which all providers propose new price policies simultaneously. Solution of the game is a Markov Perfect Equilibrium where price policy of every provider is optimal to maximize its revenue. However, they don’t consider service availability and penalty of violation, which are critical in describing consumers’ choice behavior. We argue that Markov Decision Process may not be suitable to model the game, because it is hard to predict future state of market in the subsequent steps, which is crucial to Markov Decision Process.

3. ANALYSIS ON AVAILABILITY BASED CLOUD SLA

In this section, we propose an analysis on availability based cloud SLA consisting two essential items: availability calculation and penalty calculation. We investigate SLAs of 14 well-known cloud services. Results are shown in Table 1 and 2.

3.1 Availability Calculation

Generally, availability is defined as follows:

\[
\text{Availability} = \frac{\text{ServiceTime-Downtime}}{\text{ServiceTime}} \quad (\text{Bauer and Adams, 2012})
\]

While more items are required for cloud availability calculation in practice, including:

a) Availability Commitment, to which degree cloud providers claim their service availability. Though reliability is similar with or a superset of service availability (Bauer and Adams, 2012), in all surveyed SLAs, providers prefer to take availability rate as the commitment metric. As shown in Table 1, committed availability among these cloud SLAs ranges from 99.9% to 100%, which means no more than 43.2-minute downtime per month. Ten of thirteen cloud SLAs reach up to more than 99.95%, which is 21.6-minute downtime per month. However, considering the measurement period, time granularity and exclusions, not all availability commitments are as high as expected. We will discuss it in the following paragraphs.

b) Measurement Period, the duration for cloud providers to calculate their services’ availability, usually two common forms: billing month and calendar month. The longer the measurement period is defined, the looser commitment of cloud providers make. If the measurement period is one year, cloud providers can perform unstable in a few months while stable in the rest, and still be able to fulfill the overall availability. In contrast, measurement period of one month means providers should maintain stably available services every month. According to Table 1, most cloud providers define one calendar month and one billing month as their measurement period. However, IBM Softlayer has a one-billing-year measurement period. There are two reasons why longer measurement period is not appropriate: one is that consumers’ can’t get timely penalty payment in such a long term, which will largely influence consumers’ economic interests. Another one is that long period can’t meet the flexibility and pay-as-you-go manner of current clouds. We also prefer billing month to calendar month, because billing-month measurement period is closely related with penalty payment and renewal of services.

c) Service Granularity, the scope of failed services which providers consider as unavailable (e.g., VM, host, Availability Zone). Because some failures related with hypervisor or operating system may affect a single VM rather than a rack or a whole zone, a smaller scope implies a looser criterion and a higher service quality. For instance, most providers (e.g., Aliyun Cloud) claim that any running instances suffered by downtime is counted as unavailable,
Table 1. Availability Calculation Methods of 14 Cloud Providers

<table>
<thead>
<tr>
<th>Cloud Provider</th>
<th>Availability Commitment</th>
<th>Measurement Period</th>
<th>Service Granularity</th>
<th>Time Granularity</th>
<th>Coverage</th>
<th>Exclusions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon EC2*</td>
<td>99.95%</td>
<td>billing month</td>
<td>all VMs in more than one Availability Zone (same Region)</td>
<td>1 min</td>
<td>network connection</td>
<td>(1)(3)(4)</td>
</tr>
<tr>
<td>Aliyun Cloud*</td>
<td>99.95%</td>
<td>calendar month</td>
<td>VM</td>
<td>5 min, NC</td>
<td>N/A</td>
<td>(1)(2)(3)(5)</td>
</tr>
<tr>
<td>Baidu Cloud Compute*</td>
<td>99.95%</td>
<td>calendar month</td>
<td>VM</td>
<td>5 min, NC</td>
<td>N/A</td>
<td>(1)(2)(3)(4)(5)(7)</td>
</tr>
<tr>
<td>GoGrid+</td>
<td>100%</td>
<td>billing month</td>
<td>VM</td>
<td>15 min, NC</td>
<td>hardware and hypervisor layers</td>
<td>(1)(3)(4)(5)(8)(9)</td>
</tr>
<tr>
<td>Google GCE+</td>
<td>99.95%</td>
<td>billing month</td>
<td>all VMs in more than one zones</td>
<td>5 min, NC</td>
<td>network connection, persistent disk access</td>
<td>(1)</td>
</tr>
<tr>
<td>IBM Softlayer+</td>
<td>100%</td>
<td>billing year</td>
<td>N/A</td>
<td>30 min, NC</td>
<td>network connection, consumer portal, power, HVAC</td>
<td>(1)(2)(3)</td>
</tr>
<tr>
<td>JDHosting*</td>
<td>99.9%</td>
<td>calendar month</td>
<td>VM</td>
<td>1 min, NC</td>
<td>N/A</td>
<td>(1)(3)(4)(5)(6)</td>
</tr>
<tr>
<td>Rackspace+</td>
<td>100%</td>
<td>billing month</td>
<td>N/A</td>
<td>30 min, NC</td>
<td>network connection, consumer portal, host (including hypervisor), power, HVAC</td>
<td>(1)(2)(8)</td>
</tr>
<tr>
<td>Windows Azure+*</td>
<td>99.95%</td>
<td>month</td>
<td>N/A</td>
<td>1 min</td>
<td>network connection</td>
<td>(1)(2)(3)(4)(5)(7)</td>
</tr>
<tr>
<td>Amazon EC2*</td>
<td>99.95%</td>
<td>N/A</td>
<td>VM</td>
<td>1 hour</td>
<td>N/A</td>
<td>(1)(2)(3)(4)(5)(7)</td>
</tr>
<tr>
<td>JDHosting*</td>
<td>99.95%</td>
<td>calendar month</td>
<td>VM</td>
<td>1 min, NC</td>
<td>N/A</td>
<td>(1)(2)(3)(4)(5)(7)</td>
</tr>
<tr>
<td>Rackspace+</td>
<td>1. 99.9% 2. 100%</td>
<td>month</td>
<td>N/A</td>
<td>N/A</td>
<td>Network connection, power, HVAC</td>
<td>1. (1)(4)(5)(6)(8) 2. (1)(4)(5)(6)</td>
</tr>
<tr>
<td>Telecom Cloud*</td>
<td>99.95%</td>
<td>billing month</td>
<td>at least 2 VMs in the same Availability Set</td>
<td>1 min</td>
<td>network connection</td>
<td>(3)(4)</td>
</tr>
<tr>
<td>Tencent Cloud*</td>
<td>99.92%</td>
<td>calendar month</td>
<td>VM</td>
<td>5 min, NC</td>
<td>network connection</td>
<td>(1)(3)(4)(5)(7)</td>
</tr>
</tbody>
</table>

* Chinese cloud providers
+ International cloud providers
(1) scheduled maintenance (2) scheduled upgrade (3) force majeure (4) external network failure (5) internet attack (6) host migration (7) social incident (e.g., war, political turmoil, administrative action, etc.) (8) emergency maintenance (unforeseen and needed to ensure security or reliability) (9) consumer portal
NC: if less than the time granularity, the period shall not be counted as downtime
VPS.net has two SLAs conducting two services: “with Managed Support” and “without Managed Support”

while Windows Azure has a stricter criterion, which needs more than one VM unavailable. Furthermore, Amazon EC2 and Google GCE only consider the situation as a case of unavailability when all running instances have no external connectivity in over one Availability Zone within the same Region. What’s more, some providers such as Rackspace, IBM Softlayer, Telecom Cloud and VPS.net haven’t defined their service granularity, while make a commitment of the availability in infrastructure facilities (e.g., network, power, host, HVAC, etc.). However, these commitments are indirect and immeasurable for consumers. Consumers have no idea of what incident happens with providers’ facilities except their own instances. It’s unreasonable to commit some items that can’t be measured by consumers, especially when consumers have the responsibility to report the SLA violation.
d) Time Granularity. unit downtime in the measurement period. Time granularity of most cloud providers ranges from 1 minute to 1 hour. A greater time granularity means a commitment with less actual availability rate, because it will neglect a small period of downtime. If downtime doesn’t reach the time granularity, some clouds declare these periods can’t be included into service downtime, while others would include. This means when cloud services suffer from a 7-minute downtime with a 5-minute time granularity, the eventual downtime may be 5-minute or 10-minute depending on different policies. However, Rackspace and IBM Softlayer have the 2nd longest time granularity, which is NC. This means their 100% availability is not real. When downtime comes to 29 minutes, it won’t be counted into downtime or compensated eventually. If we define one minute as a standard time granularity, their actual availability commitment is 99.93% (1-30mins/30days), which means their availability commitments are different from what they guarantee.

<table>
<thead>
<tr>
<th>Cloud Provider</th>
<th>Service Credit (Penalty)</th>
<th>Calculation Methods</th>
<th>Penalty Cap</th>
<th>Payment Method</th>
<th>Report Onus</th>
<th>Report time restriction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon EC2*</td>
<td>&lt;99.95% 10% &lt;99% 30%</td>
<td>Ratio of Total Charge</td>
<td>N/A</td>
<td>future payment</td>
<td>consumer</td>
<td>N/A</td>
</tr>
<tr>
<td>Aliyun Cloud*</td>
<td>100*downtime</td>
<td>Ratio of Downtime</td>
<td>100%</td>
<td>future payment</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Baidu Cloud Compute*</td>
<td>100*downtime</td>
<td>Ratio of Downtime</td>
<td>100%</td>
<td>future payment</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>GoGrid+</td>
<td>100*downtime</td>
<td>Ratio of Downtime</td>
<td>100%</td>
<td>future payment</td>
<td>consumer</td>
<td>within 2 days</td>
</tr>
<tr>
<td>Google GCE+</td>
<td>&lt;99.95% 10% &lt;99% 25% &lt;95% 50%</td>
<td>Ratio of Total Charge</td>
<td>50%</td>
<td>future payment</td>
<td>consumer</td>
<td>N/A</td>
</tr>
<tr>
<td>IBM Softlayer+</td>
<td>5% of the fees for each 30-minute downtime</td>
<td>Ratio of Downtime</td>
<td>N/A</td>
<td>future payment</td>
<td>consumer</td>
<td>within 7 days</td>
</tr>
<tr>
<td>JDHosting*</td>
<td>1 day for 1-minute downtime</td>
<td>Ratio of Downtime</td>
<td>200%</td>
<td>future payment</td>
<td>(90 days available)</td>
<td>N/A</td>
</tr>
<tr>
<td>Rackspace+</td>
<td>1. Data Plane: 5% of the fees for each 30-minute downtime 2. Control Plane: &lt;99.9% 10% &lt;99.5% 20% &lt;99% 30%</td>
<td>1. Ratio of Downtime 2. Ratio of Total Charge</td>
<td>100%</td>
<td>future payment</td>
<td>consumer</td>
<td>within 30 days</td>
</tr>
<tr>
<td>Telecom Cloud*</td>
<td>&lt;99.95% 24 hrs &lt;99% 720 hrs</td>
<td>Fixed Value</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Tencent Cloud*</td>
<td>100*downtime</td>
<td>Ratio of Downtime</td>
<td>100%</td>
<td>future payment</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ucloud*</td>
<td>1*downtime</td>
<td>Ratio of Downtime</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>VPS.net +</td>
<td>10*downtime</td>
<td>Ratio of Downtime</td>
<td>100%</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Windows Azure**</td>
<td>&lt;99.95% 10% &lt;99% 25%</td>
<td>Ratio of Total Charge</td>
<td>N/A</td>
<td>future payment</td>
<td>consumer</td>
<td>before the end of next billing month</td>
</tr>
<tr>
<td>360Kaopu Cloud*</td>
<td>&lt;99.92% 10% &lt;99% 25%</td>
<td>Ratio of Total Charge</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
e) **Coverage.** Cloud service is assumed as unavailable when components in the coverage fail. In most cases, cloud providers view availability equivalent to internet connectivity of a running instance. However, in practice, internet connectivity of instance is one of prerequisites for instance availability, failures caused by hypervisor, CPU, RAM, or operating system can also lead to instance unavailability. Moreover, considering the SLA of whole cloud services is evaluated, not only the running instances, but other relevant services like cloud management (e.g., using dashboard to create, destroy instances) should be taken into availability as well. On one hand, most clouds leverage only network connection to detect the availability status of a running instance, only a minority of cloud providers (e.g., Google GCE) includes other elements in the coverage. On the other hand, availability commitments for relevant cloud services are missing in current cloud SLAs. For example, only Rackspace and IBM Softlayer ensure the availability of consumer portal, where consumers manage their instances and deploy their applications. GoGrid explicitly introduces consumer portal into its exclusions. IBM Softlayer guarantees 100% availability of Customer Portal, which, however, is included into its instance service. Moreover, some cloud SLAs haven’t clear coverage of their availability commitments, which is impractical for counting penalty. Most Chinese clouds expose their drawback in this aspect.

f) **Exclusions**, the situations can’t be counted as unavailability. As presented in Table 1, all cloud SLAs excludes period of scheduled maintenance and upgrade from availability calculation. During the scheduled maintenance and upgrade, instances are unavailable to consumers, so actual availability commitment is lower than the claimed one in every cloud SLA. The gap between actual and committed availability depends on the frequency and duration of maintenance for each cloud. Even with a warning, consumers could suffer loss of business from the maintenance and upgrade. IBM Softlayer takes methods to make up consumers’ loss by guaranteeing that hardware replacement and upgrade of cloud servers shall be completed within 2 hours. Most cloud SLAs also exclude some incidents outside of providers’ reasonable control, including force majeure, social incidents and external network failure. For instance, external network failures which bring enormous loss to consumers are partially under providers’ control. Cloud providers can have a few alternatives on external network, for example, selecting a high-quality ISP, or building a reliable network themselves or preparing a backup network for emergency. Moreover, external network failure will bring enormous loss to consumers and internet attack is a similar issue. Cloud providers should not be blamed for internet attack, but it’s better for them to undertake responsibility for preventing cloud servers from hackers’ attacks.

3.2 Penalty Calculation

When the cloud services cannot meet availability commitment claimed in cloud SLA, providers would pay back penalty in the form of service credits to compensate for the violation. There are three essential items for calculating the penalty: Calculation Method, Penalty Cap and Payment Method.

a) **Calculation Method.** There are three methods to count the penalty, all of which are based on different violation level (i.e., unavailability rate or downtime): 1. **Method Ratio of Total Charge:** A certain percentage of total charges paid to consumers in the measurement period based on different violation levels; 2. **Method Fixed Value:** fixed value at different violation levels; 3. **Method Ratio of Downtime:** A certain ratio of downtime, usually greater than 1.

b) **Penalty Cap.** It is the maximal value of penalty that providers pay back for violation. The method of Ratio of Total Charge and Fixed Value have inherently penalty cap, while the method of Ratio of Downtime will set up a maximal penalty value as the penalty cap.

c) **Payment Method.** Most cloud providers claim the penalty can only be paid as future payments in the same service. No real money is paid back to consumers’ accounts.

Besides, though consumers may not be the best part to report violation, most international cloud providers claim in their SLAs that the responsibility of reporting violation belongs to the cloud consumers. That means, consumers are the reporting onus and have to detect violation, that is, consumers need to report server logs or monitoring data within a time restriction. If they haven’t detected violation of the cloud service and report it to the provider in a certain period, they can’t get the compensation. Moreover, some providers would punish those consumers who report a false violation (e.g., IBM Softlayer). Meanwhile, all of the surveyed Chinese cloud providers haven’t defined clearly the onus to report violation yet, which may raise the problem of unclear responsibility.

Because all cloud providers apply the penalty payment only to future payment for their services, we calculate the payment in the form of service credit percentage to measure the penalty paid cloud providers when violation occurs. Here we define penalty degree as ratio of penalty cloud provider should pay to consumers according to availability rate. As shown in Table 2, penalty degrees are largely different among various cloud providers. Some providers won’t pay much for even a long period of downtime (e.g., Ucloud, VPN.net), while some pay almost as much as the consumers’ income for penalty (e.g., Aliyun Cloud, Tencent Cloud, GoGrid). JDHosting pays more than the income, which is considered as a punitive compensation.

Inherently associated with cloud providers’ availability and financial management (e.g., price and cost of services), penalty degree plays an essential role in cloud SLA and will influence the decision of cloud consumers when choosing cloud providers. Thus we propose an availability penalty model for cloud providers to define their penalty strategies.
in their SLAs in a competitive market to attract consumers and achieve maximal profits.

4. A COMPETITIVE PENALTY MODEL FOR AVAILABILITY BASED CLOUD SLA

In the section, we discuss profits of cloud providers and cloud consumers, and then propose a Penalty Based Profit Maximization Algorithm to formulate the game procedure and calculate optimal penalty degrees for cloud providers. Table 3 presents all the mathematical notation used throughout this paper.

Assuming there are n cloud providers and m cloud consumers in the market, to maximize profit, each cloud provider and cloud consumer make best decision for themselves. We adapt runtime SLA negotiation instead of a broker-based process for two reasons. On one hand, runtime SLA negotiation is a win-win solution for both cloud provider and consumer. On the other hand, it is also easy to realign the service and reduce the cost (Hani et al., 2015).

When selling resources to the consumers, cloud providers make optimal penalty strategies, so as to compete with other providers in the competitive market and achieve maximal profit. Cloud consumers choose the provider who best suits their economic and availability requirements. The competition among cloud providers is formulated as a non-cooperative game where cloud providers compete with each other through lower resource price, higher availability or stricter penalty strategy and makes decision simultaneously and independently to achieve maximal profit (Brandenburger, 1992). In a competitive market, cloud consumers are free to choose resources according to the cost and availability requirement and make greatest profit by selecting suitable resources from different cloud providers. This game procedure lasts until all cloud providers and cloud consumers reach an agreement finally.

4.1 Profit of Cloud Providers

The profit of cloud providers is defined as follows and the goal is to maximize it:

\[ \text{max Profit} = \text{Revenue} - \text{Cost} - \text{Penalty}. \] (1)

If Consumer j chooses Provider i, then the revenue of Provider i earned from Consumer j is:

\[ \text{Revenue}_{ij} = p_i \eta_j, \]

where \( p_i (1 \leq i \leq n) \) indicates the unit price of resources provided by Provider i and \( \eta_j (1 \leq j \leq m) \) indicates the number of resources requested by Consumer j.

The unit cost of resources provided by Provider i is denoted as \( C_i \). Assuming number of reserved instances is a ratio (denoted as \( \varphi_i \)) of request number for Provider i, which is related with service availability of Provider i. Resource cost is proportional to number of requests. Therefore, to satisfy resource requests of Consumer j, the cost of Provider i is defined as:

\[ \text{Cost}_{ij} = C_i \eta_j (1 + \varphi_i). \]

Once providers can’t guarantee service availability committed in their SLAs, they will pay penalty to consumers to make compensation for their economic loss. Since most of the cloud providers choose Ratio of Downtime as their penalty calculation method, as showed in Table 2, we define the penalty as a fixed ratio of downtime, which is called Penalty Degree and denoted as \( k_i \) for Provider i. Thus the penalty that the Provider i pays back to Consumer j is presented as:

\[ \text{Penalty}_{ij} = \text{Revenue}_{ij} \cdot k_i (1 - A_i), \]

where \( A_i \) is the service availability of Provider i. We assume \( A_i \) follows Beta distribution. Beta distribution is a common model for the random behavior of percentages and proportions. It is a continuous probability distribution defined on the interval \([0, 1]\) and parameterized by two positive parameters, \( \alpha \) and \( \beta \). Mean and variance of Beta distribution is equal to \( \frac{\alpha}{\alpha+\beta} \) and \( \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \) respectively.

Therefore, the formulation of \( A_i \) is:

\[ A_i \sim \text{Beta}(\alpha_i, \beta_i). \]

Generally, more reserved instance will lead to higher service availability, because faulty instance can be quickly replaced by reserved instance based on techniques like fault tolerance, protective redundancy and load balancing (Nabi et al., 2015). Hence, we assume reserved instance ratio of Provider i follows the following equation:

\[ \varphi_i = \frac{1}{\varphi_o} \cdot \log \left( 1 - \frac{a}{a + \beta} \right). \]

where $\frac{1}{\varphi_0}$ is a coefficient for $\varphi_i$. In practice, $\varphi_0 = \log(1 - \hat{A})$, where $\hat{A}$ is the availability rate of service without reserved instance.

Then expected profit of Provider $i$ is counted as follows:

$$ E[\text{Profit}_i] = E[\text{Revenue}_i - \text{Cost}_i - \text{Penalty}_i] $$

$$ = \sum_{j=1}^{m} (\text{Revenue}_{ij} - \text{Cost}_i - \text{Penalty}_i) q_{ij} $$

$$ = \sum_{j=1}^{m} \left( p_i r_j - C_i r_j (1 + \varphi_j) - p_i r_j k_i (1 - \hat{A}_i) \right) q_{ij} $$

where $q_{ij}$ is the probability of Consumer $j$ purchasing service provided by Provider $i$. (choice probability is described in Subsection 4.3).

As a result, the main purpose becomes to find an optimal penalty degree $k_i$ as a penalty strategy to maximize the profit of Provider $i$:

$$ k_i = \arg \max E[\text{Profit}_i], \quad (3) $$

### 4.2 Profit of Cloud Consumers

The profit of a cloud consumer is consisted of its revenue, cost and penalty of availability violation (paid by the cloud provider). The profit of a cloud consumer $\text{Profit}^*$ is calculated as:

$$ \text{Profit}^* = \text{Revenue} - \text{Cost} + \text{Penalty}. $$

Cloud consumers utilize resources offered by cloud providers to run their own businesses or applications. Usually, the more end user requests, which leads to more revenue, the more resources will be required. In other words, the revenue can be calculated based on the resources used and their availability. That is to say, the revenue of Consumer $j$ is proportional to both the number of consumed resources and their availability. $R_j^*$ is assumed as the unit revenue of Consumer $j$. If Consumer $j$ chooses Provider $i$ with availability $A_i$, the revenue gained by Consumer $j$ $\text{Revenue}^*_j$, is defined as:

$$ \text{Revenue}^*_j = R_j^* r_j A_i. $$

Once Consumer $j$ choose Provider $i$, the cost of Consumer $j$ is equal to the revenue of Provider $i$, which is $\text{Revenue}_i$.

According to availability penalty strategy defined in SLA, if the cloud provider cannot meet their availability commitment defined in SLA, Consumer $j$ would gain compensation from Provider $i$ which is $\text{Penalty}_ij$. Therefore, the profit of Consumer $j$ is:

$$ \text{Profit}^*_ij = \text{Revenue}^*_j - \text{Revenue}_i + \text{Penalty}_ij $$

$$ = R_j^* r_j A_i - p_i r_j + p_i r_j k_i (1 - A_i) $$

$$ = -p_i r_j (1 - k_i) + (R_j^* r_j - p_i r_j k_i) A_i $$

### 4.3 Cloud Consumers’ Choice Probability
We present a method to calculate the probability of Consumer $j$ choosing Provider $i$ according to the providers’ unit price of resources, availability probability and penalty strategy.

Let $B_{ij} = -p_i r_j (1 - k_i)$ and $C_{ij} = R_i r_j - p_i r_j k_i$, then cloud consumer’s profit (Equation 4) becomes:

$$\text{Profit}^*_i = B_{ij} + C_{ij} A_i.$$

As described above, $A_i$ (service availability of Provider $i$) is a random variable following Beta distribution. We denote the probability density function (PDF) and cumulative function (CDF) of Beta distribution as $f(x; \alpha, \beta)$ and $F(x; \alpha, \beta)$ respectively.

The cloud consumer will choose the best offer from cloud providers, which means Consumer $j$ will choose Provider $i$, only if it can make more profits from Provider $i$ than choosing other providers. Therefore, the probability of Consumer $j$ choosing Provider $i$ (denoted as $q_{ij}$) is calculated as follows:

$$q_{ij} = \text{Prob} \left( \text{Profit}^*_i > \text{Profit}^*_{i'}, \forall i' \neq i \right)$$

$$= \text{Prob} \left( B_{ij} + C_{ij} A_i > B_{ij'} + C_{ij'} A_{i'}, \forall i' \neq i \right)$$

$$= \text{Prob} \left( A_i - \frac{B_{ij'}}{C_{ij'}} > A_i - \frac{B_{ij}}{C_{ij}} \right)$$

$$= \text{Prob} \left( A_i > \frac{B_{ij'}}{C_{ij'}} + \frac{B_{ij}}{C_{ij}} \right)$$

Based on the probability density function $f(x; \alpha, \beta)$ and the cumulative distribution function $F(x; \alpha, \beta)$, some algebraic operations lead to the following equation:

$$q_{ij} = \int_0^1 \prod_{i' \neq i} F \left( \frac{B_{ij'} + C_{ij'} A_{i'} - B_{ij}}{C_{ij'}} \right) f(A_i) dA_i \quad (5)$$

Therefore, $q_{ij}$ is a function of $k_i$. In a further step, $E[\text{Profit}^*_i]$ becomes a function of $k_i$ (Equation 2). Cloud provider can optimize their expected profit by selecting a certain penalty strategy $k_i$.

4.4 A Penalty Based Profit Maximization Algorithm

We focus on availability based cloud SLAs in the negotiation. After all cloud consumers make selection according to the current SLAs, cloud providers can adjust their penalty strategies, in the scope of penalty cap, to find an optimal penalty degree $k_i$ which can maximize cloud providers’ profits. Based on cloud providers’ new SLA commitments, cloud consumers can also adjust their selection. Then the cloud providers can change their penalty strategies based on the new choice of cloud consumers again. This procedure continues until no provider can increase his revenue and profit by altering penalty strategy, then providers and consumers make a deal eventually.

Algorithm 1 presents the Penalty Based Profit Maximization Algorithm, and the whole game procedure is presented as follows:

**Step 1.** At initial Stage ($t = 0$), all providers propose their resource price $p_i$, initial penalty strategies $k^0_i$ and calculate its reserved instance ratio $\varphi_i$ for high availability.

**Step 2.** At Stage $t$, according to every provider’s price $p_i$ and current penalty degree $k^t_i$, consumers choose the provider that can maximize their profits. For uncertainty of availability, their choices are presented as a probability $q^t_{ij}$ (Equation 5). $t$ indicates the index of stage.

**Step 3.** Following all consumers’ choices, cloud providers may find a better penalty ratio to maximize their profits $\text{Profit}^*_i$ and set this ratio as $k^{t+1}_i$. Note that all providers will propose their penalty strategies simultaneously and independently so their strategies are unknown to each other when making decision.

**Step 4.** The procedure repeats from Step 2 to Step 3 until market state reaches steady state. Equation 6 presents the stopping criterion:

$$|k^{t+1}_i - k^t_i| < \epsilon \quad (6)$$

where $\epsilon$ is a parameter of minimal variation of penalty degree between two neighbor rounds of negotiation. In this state, no provider will alter his penalty strategy. If the algorithm hasn’t reached a stable state denoted in Equation 6 until the number of iteration exceeds maximum times (denoted as $N$), the whole procedure will go back to Step 1 with new initial penalty strategy $k^0_i$ and try to achieve a new convergence state.

5. Evaluation of the Algorithm

In this section, we first evaluate the Penalty Based Profit Maximization Algorithm with real data of public cloud providers, and then conduct a sensitive analysis of the algorithm.

5.1 Simulation Based Evaluation on Public Cloud Providers

We select five cloud providers in China: Aliyun Cloud, Tencent Cloud, Baidu Cloud Compute, Ucloud and JDHosting, and apply our algorithm to analyze their SLA penalty strategies. Considering that the unit price of cloud service is different among different countries and geographic locations, we don’t include international cloud providers in the evaluation. It’s interesting that the current SLA penalty degrees among five cloud providers are quite different to each other. It may be due to the fierce competitive and rapidly growing market in China, and main objective of some providers is to attract as many consumers as possible, instead of profit.

Price and penalty degree in cloud SLA of the five cloud providers are listed in the Table 4. Because the cost of each cloud provider is not public, we assume the cost is half of the service price (which can be replaced of real cost to get more accurate simulating results). Empirical parameter in reserved instance ratio $\overline{A}$ is 0.7, thus $q^0 = -1.2040$. We also assume the real availability rate for each provider following Beta distribution with parameter $\alpha$ and $\beta$. We set $\alpha = 10$, and $\beta = 2$, which means their average availability rate is 83.33%. And resource requests of each consumer are
assumed to 100 unit of cloud services. Thus total providers’ sale volume is 400. Actually, the penalty degree results are independent of requests of cloud consumers, which will be proved in the later sensitivity analysis.

The results of evaluation are listed in Table 4. Figure 1 and 2 elaborate the negotiation process (iterations in the algorithm) among five cloud providers in view of cloud providers’ profit and penalty degree.

Figure 1 and Figure 2 show that it takes 61 iterations to reach stability in the algorithm. However, the value of profit and penalty degree don’t change much after the 15th iteration, which means the negotiation doesn’t last too long for cloud providers and cloud consumers. People can make an agreement and close the deal very soon. We also find that cloud providers compete with each other during negotiations by increasing their penalty degrees so as to get more consumers, and their profits decrease in the meanwhile.

The result of the evaluation (Table 4) shows that currently Ucloud sets their penalty strategy close to the optimal point, while others’ penalty degrees (Aliyun, Tencent Cloud, and JDHosting) are quite greater than their optimal penalty degree. Because of high unit price and optimal penalty degree, no consumers will choose JDHosting, and its sale volumes and revenues drop to zero. Based on the assumption of the algorithm, those optimal penalty degrees are suggested in a mature market.

### 5.2 Sensitivity Analysis of the Penalty Based Profit Maximization Algorithm

<table>
<thead>
<tr>
<th>Provider</th>
<th>Unit Price*</th>
<th>Unit Cost</th>
<th>Current Penalty Degree *</th>
<th>Optimal Penalty Degree</th>
<th>Simulated Provider’s Sale Volume</th>
<th>Simulated Provider’s Profit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aliyun Cloud</td>
<td>198</td>
<td>99</td>
<td>100</td>
<td>1.4918</td>
<td>115.72</td>
<td>3180.3</td>
</tr>
<tr>
<td>Tencent Cloud</td>
<td>200</td>
<td>100</td>
<td>100</td>
<td>1.5261</td>
<td>106.32</td>
<td>2891.4</td>
</tr>
<tr>
<td>Baidu Cloud Compute</td>
<td>190</td>
<td>95</td>
<td>100</td>
<td>1.3251</td>
<td>148.04</td>
<td>4271.4</td>
</tr>
<tr>
<td>Ucloud</td>
<td>218</td>
<td>109</td>
<td>1</td>
<td>1.7359</td>
<td>29.92</td>
<td>740.2</td>
</tr>
<tr>
<td>JDHosting</td>
<td>290</td>
<td>195</td>
<td>100</td>
<td>1.6538</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

* Unit price and Penalty degree are obtained from cloud providers in April 2015.
According to the Penalty Based Profit Maximization Algorithm, we conduct a sensitive analysis to find out the main factors in the algorithm that have influence on the optimal penalty strategy of cloud providers. In Table 5, we present following five inputs of the algorithm to evaluate their impact on the result: consumer’s request, initial penalty parameter in Consumers’ Requests, Initial Penalty Degree, cloud provider’s cost, cloud resource price and real availability.

In the sensitive analysis, we assume that there are three cloud providers and four consumers in the competitive market, where each consumer would choose the best offer from three cloud providers. Thus cloud providers adjust

<table>
<thead>
<tr>
<th>Table 5. Sensitivity Analysis of the Penalty Based Profit Maximization Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Group</strong></td>
</tr>
<tr>
<td><strong>Control Group</strong></td>
</tr>
<tr>
<td><strong>Consumers’ Requests</strong></td>
</tr>
<tr>
<td><strong>Initial Penalty Degree</strong></td>
</tr>
<tr>
<td><strong>Unit Cost</strong></td>
</tr>
<tr>
<td><strong>Unit Price</strong></td>
</tr>
<tr>
<td><strong>Real Availability Parameter α</strong></td>
</tr>
</tbody>
</table>
their penalty strategies continuously to meet consumers’ requests and maximize their revenues. After several iterations, both sides reach an agreement and make the final deal. In the simulation, unit price, unit cost, initial penalty degree, consumers’ requests, real availability probability, reserved instance ratio are set differently to evaluate the algorithm. We also assume the real availability rate for each provider following Beta distribution with parameter $a$ and $\beta$. To simplify the simulation, $\beta$ is set as a fixed value 2. $a$ is set differently to meet average availability rate of each cloud provider. In the simulation, $a$ belongs to $\{5, 10, 15\}$. This means availability rate in the simulation are 71.43%, 83.33%, and 88.24%, and variances of availability rate are 0.0255, 0.0107, and 0.0058. Empirical parameter in reserved instance ratio $\lambda$ is set as $\{0.6, 0.7, 0.8\}$.

Here we make a few assumptions about parameters in the algorithm: maximum iteration $N$ is 1000, and stopping criterion is set as $\epsilon = 10^{-8}$. When meeting the stopping criterion, cloud providers and cloud consumers make an agreement and the game procedure ends.

First group (id = 1) is a control group, which will be compared with following groups.

In the second group of simulations (id = 2, 3), we aim to validate the irrelevance between cloud consumers’ requests and cloud providers’ strategies. The results of simulations show the consumers’ requests don’t affect the final penalty degree. And the increased requests will yield a proportional revenue of cloud providers finally. This result is reasonable for our assumption and algorithm.

The third group (id = 4, 5, 6) shows that whatever the initial penalty degree is, it won’t have influence on the final results of optimal penalty degree and revenue, unless it results in the divergence. The 6th simulation illustrates an exception. An extremely large initial penalty degree would result in divergence in the final results.

The fourth group (id = 7, 8) reveals that different costs for cloud providers lead to different penalty strategies. With the increase of cost, cloud providers will lower their penalty degree to gain more revenue. Moreover, with the same unit price, the higher cost becomes, the smaller revenue the provider owns, which is reasonable from the perspective of economics.

Different prices offered by cloud providers are set in the fifth group (id = 9, 10). The higher price will lead to a higher penalty degree to attract consumers through promising more compensation for SLA violation. However, higher price doesn’t mean higher revenue in the end. Because the revenue goes down if no one buys cloud services due to the expensive price and the market share will also be taken by other providers in the race condition.

Cloud availability is essential to both cloud providers and cloud consumers. The sixth group (id = 11, 12) validates the impact of availability on each cloud provider. According to the results, a higher $a$ value, which indicates a higher cloud availability, will lead to a higher penalty degree to convince consumers, which will attract more consumers and bring greater revenue in the end.

Reserved instance ratio $q_i$ is a main factor of cloud provider’s resource cost. From the seventh group (id = 13, 14), higher reserved instance ratio $q_i$ (or smaller $\lambda$ ) leads to higher cost, which will lower cloud providers’ profits and increase the penalty degree.

From comparison of the control group and other groups (e.g., id = 7, 8, 9, 10, 11, 12, 13, 14), the number of iteration decreases largely when difference of parameters in the algorithm is large. It indicates when cloud providers offer different prices or have different cost, it takes a short time to make a decision in the negotiation and the algorithm converges to the optimal penalty strategy and maximal revenues quickly. However, if all parameters of cloud providers are the same, it will last long to negotiate and the final results of each cloud provider will become the same as well.

To conclude the sensitive analysis, unit price of cloud resource, unit cost and real cloud availability are three main factors in the Penalty Based Profit Maximization Algorithm. Penalty degree will increase with the drop of cost or the rise of unit price or real availability.

6. CONCLUSIONS AND FUTURE WORK

Lacking of a formal SLA and a guidance for cloud providers to make suitable penalty strategies in the SLA prompt us to build the competitive penalty model. In this paper, we summarize the methods of availability and penalty calculation among popular cloud providers. We describe behavior of cloud providers and cloud consumers as a game in a competitive penalty model. Based on the methods and game procedure, we propose a Penalty Based Profit Maximization Algorithm to achieve maximal profit for each cloud provider. This algorithm is capable of guiding popular cloud providers to set an optimal penalty degree in their cloud SLAs.

The proposed model has several limitations. For instance, we haven’t considered the case that a new provider enters the market during game procedure. We also assume all cloud instance is identical. In the future work, we will include more variables such as claimed availability rate in the cloud SLAs, different downtime impact of cloud consumers and so on. We are going to collect more real-world data (i.e., resource cost, real availability of cloud service) to validate the algorithm. We believe with more real features our work will make cloud providers convenient when making decisions on their penalty strategies.

7. ACKNOWLEDGEMENT

The project is supported by National Natural Science Foundation of China (61232005) and National High Technology R&D Program of China (2015AA016009).
8. References


Cloud or Fog? The business realities of cloud computing for UK enterprises, NTT Commm., 2009


JDHosting SLA, http://hosting.jd.com/hostingIntro/user

Ucloud SLA, http://www.ucloud.cn/pact/uhost_sla


Google Compute Engine Service Level Agreement (SLA), https://cloud.google.com/compute/sla


VPS.Net Terms of Service, http://vps.net/about-vps-net/terms-service


TeleCom Cloud SLA, http://www.ctyun.cn/terms


Authors

Xiaoyong Yuan is currently a Ph.D student in Computer & Information Science & Engineering Department, University of Florida, USA. He received his BS degree in mathematics from Fudan University and MS degree in computer engineering from Peking University in China. His research interests lie in cloud computing and deep learning systems.

Hongyan Tang received the B.S. degree in software engineering from Software College, Xiamen University, China, in 2014. She is currently a graduate student in the school of Software and Microelectronics, Peking University, China. Her research interests include cloud computing, big data analysis and dependability of distributed system.

Ying Li is currently a professor in the school of Software and Microelectronics, Peking University, China. From 2001 to 2012, she worked as a research staff member in IBM China Research Laboratory, where she was also a senior manager of the department of distributed computing and system management. Her research interests are distributed computing particularly in Web services, cloud computing, and distributed machine learning platform. She published over 40 journal and conference papers, and filed 30 patents in distributed computing area. She has participated in many industrial projects and made solid contribution to commercial software systems. She served as PC member of several international conferences and reviewer of international journals.

Tong Jia is currently a Ph.D. student in school of Software and Microelectronics, Peking University. From 2014 to 2015, he has published 4 papers as a coauthor. His research interests include cloud computing and big data, software availability and reliability.

Tiancheng Liu is currently Research Staff Member in IBM Research – China. He is focusing on developing system management solutions and techniques in distributed environment, e.g. Internet of Things (IoT) and Cloud. The primary ongoing projects include configuration automation in Cloud and data center, software defined networking and OpenStack operations. In the past, He also worked on Cloud Migration to automate the application migration to IBM Cloud and IoT device management. Before playing this role, he worked on several research projects related to collaboratively manage IT system from remote, Java virtualization and resource management. He led two ExtremeBlue project related to cost analysis in Cloud in 2009, and 3D Internet and the virtual world in 2007. Tiancheng Liu received the Bachelor in 2000 and Ph.D in 2006 in Computer Science from Peking University, and joint IBM right after.

Zhonghai Wu is a professor in the school of Software and Microelectronics, Peking University, China. His research fields include cloud computing, embedded system and information security. He received his Ph.D. of computer science from Zhejiang University. He published over 80 journal and conference papers, and held more than 20 patents. He served as a council member of CCF and TC chair of Embedded System.