Applied Mathematics & Information Sciences
An International Journal

An Efficient and Secure Cloud-Based Distributed Simulation System

Heng He[1,2], *Ruixuan Li*[1], *Xinhua Dong*[1], *Zhi Zhang*[2] *and Hongmu Han*[1]

[1] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
[2] School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, China

Received: Feb. 15, 2012; Revised Apr. 20, 2012; Accepted May 5, 2012
Published online: 1 August 2012

Abstract: For the deficiency of High Level Architecture (HLA), it is not well suited for large-scale distributed simulation systems. To enhance the capability of HLA and satisfy the new requirements of large-scale distributed simulations, an efficient and secure cloud-based distributed simulation system, namely CDS, based on the cloud computing technology and HLA is proposed. CDS provides a service-oriented simulation support environment. It can provide users with on-demand simulation resources, run simulations on the wide area network efficiently, realize share and reuse of simulation resources, improve load balancing capability of the simulation, and provide security guarantee. Compared with previous simulation grid systems, CDS is efficient, secure, universal and practical. From the view of technical implementation, key technologies of CDS are described in details. These technologies include: (1) an agent-based invoking and callback strategy in simulation run-time infrastructure, (2) resource management and scheduling schemes in management center, (3) an efficient security scheme. Through developing CDS and a practical simulation application on it, the performance of CDS is evaluated and the results demonstrate the proposed system functionalities.

Keywords: Cloud computing, HLA, Simulation grid, Identity-based cryptography.

1. Introduction

The High Level Architecture (HLA) is a U.S.DoD-wide initiative to provide architecture for support interoperability and reusability of all types of models for distributed interactive simulations. HLA has been adopted by IEEE as an open standard, the IEEE 1516 [1]. It has experienced tremendous growth over the past few years, playing an increasingly critical role in all areas of science and engineering. However, with the development of large-scale, fine-granularity and long-time distributed simulation applications, HLA cannot be well suited for these simulations gradually for its inefficient utilization of simulation resources, lack of load balancing capability, weak fault tolerance capability, and complicated simulation deployment process. Recent studies show that a "model-centered organization mode" is required for the efficient execution of these advanced simulations [2], which makes the simulation models fixed, organizes simulations by providing models on demand and runs simulations on the wide area network efficiently, also provides security guarantee for the simulations. However, HLA can not satisfy the new requirements.

Simulation grid technology [3] has provided a considerable solution. Many domestic and foreign institutions have done a lot of relevant researches. Some well-known projects include CrossGrid [4], DS-Grid [5], Federation Grid [6] and Simulation Grid [3]. However, most of these researches are in the scope of concept study, architecture and prototype system design, or in the specialized fields. In our prior work [7], we proposed a grid-based distributed simulation platform for large-scale simulations, which can be further enhanced in the aspects of security and efficiency. Thus, it's necessary to do research on more universal and practical technologies for large-scale distributed simulation systems.

Cloud computing [8,9] is becoming a hot research field of information science. It provides an advanced computing platform in which network users can access computing services on demand anytime and anywhere. Integrating HLA with cloud computing technology, we propose an ef-

* Corresponding author: e-mail: willam1981@gmail.com

ficient and secure Cloud-based Distributed Simulation system (CDS), which can enhance the capability of HLA and satisfy the new requirements of large-scale simulations effectively. It can provide users with on-demand simulation resources, run simulations on the wide area network efficiently, realize share and reuse of simulation resources, improve load balancing capability of the simulation, and provide security guarantee. Comparing with the existing simulation grid systems, CDS is efficient, secure, universal and practical. In the rest of the paper, we first introduce some preliminaries related to the proposed research, and then describe the architecture of CDS and its key technologies in details. At last we evaluate the system performance.

2. Preliminaries

2.1. High Level Architecture

The HLA is a standard architecture upon which one may design and integrate application-specific simulators. Kuhl et al. [10] succinctly describe it as "... the glue that allows you to combine computer simulators into a larger simulation." The HLA consists of a set of rules to which all compliant applications must adhere, a common model for sharing data and documenting shared data, and an application program interface (API) for a Runtime Infrastructure (RTI) software system that integrates the individual simulators. RTI is the software that implements the HLA interface specification and runs the simulation. It contains six major types of functions, based on which it can provide mechanisms for managing time and data, and synchronizing the interplaying simulators. The effort required to integrate HLA-compliant simulators into larger simulation systems is far less than that of integrating simulators that are not designed in accordance with HLA specifications. More details of HLA and RTI can be found in [10].

In the HLA simulation, each individual simulator is named as a federate, which the HLA integrates together into a collaborative simulation. All federates in the simulation form a federation.

2.2. Cloud computing

Cloud computing is the newly born Internet-based computing technique which integrates, optimizes and provides computing ability, aiming to simplify the clients' computing jobs by the way of renting resources and services [8,9]. Currently, there is no recognized authoritative definition of cloud computing, for the agreement of the universal protocol of cloud computing has not been reached. Generally, cloud computing is considered as the computing which is based on the Internet and offers resources to the clients in virtualized, convenient, efficient, and underlying-process-acknowledgement-free ways.

Grid computing is also a popular distributed computing technology which integrates resources to solve problems over the Internet. Cloud computing is considered to be the technology derived from grid computing with different problem-dealing-aspects of processing but core concept unchanged. The paramount notion of grid computing is collecting all available resources to solve problems (mainly scientific ones) that individual computing center cannot deal with. The most famous case of grid computing, SETI@HOME, searches for extra terrestrial intelligence using the spare resources from the volunteer computers connected to Internet all over the world. On the other hand, cloud computing aims to supply efficient, qualified services with part of the available large scale of computing resources.

Cloud computing has many advantages in cost reduction, resource sharing, time saving for new service deployment. While in a cloud computing system, most data and software that users use reside on the Internet, which bring some new challenges for the system, especially security and privacy. Since each application may use resource from multiple servers. The servers are potentially based at multiple locations and the services provided by the cloud may use different infrastructures across organizations. All these characteristics of cloud computing make it complicated to provide security in cloud computing.

2.3. Identity-based cryptography

Identity-based cryptographic (IBC) scheme [11] is a kind of public-key based approach, in which user's identity is used as the public key. Compared with traditional public-key systems, such as PKI, IBC has some advantages, especially for large-scale distributed applications. It can simplify the key management complexity significantly. The encryption and decryption can be conducted offline without the key generation center (KGC). Hierarchical identity-based cryptography (HIBC) has been proposed in [12] to improve the scalability of the standard IBC scheme, in which multiple KGCs are used to cooperatively allocate hierarchical identities to users in their domains. Recently IBC and HIBC have been proposed to provide security for some Internet applications. For example, applying IBC in the grid computing and cloud computing security has been explored in [13,14]. In the following, we briefly introduce the mathematical concepts that are required in this paper.

A. Bilinear pairings

Bilinear pairings are the most commonly used primitives in IBC and HIBC.

Let G_1 be an additive cyclic group of large prime order q, G_2 be a multiplicative cyclic group of the same order and P be a generator of G_1. A cryptographic bilinear map e is defined as $e : G_1 \times G_1 \rightarrow G_2$ with the following properties:

Bilinear: $e(aR, bS) = e(R, S)^{ab}$ $\forall R, S \in G_1$ and $a, b \in Z_q^*$.

Non-degeneracy: For each $R \neq O \in G_1$, there exists $S \in G_1$ such that $e(R, S) \neq 1$, where O is the identity element in G_1 and 1 is the identity element in G_2.

Computable: There exists an efficient algorithm to compute $e(R, S)\ \forall R, S \in G_1$.

In general implementation, G_1 is the group of points on an elliptic curve and G_2 denotes a multiplicative subgroup of a finite field. Typically, the mapping e is derived from either the Weil or the Tate pairing on an elliptic curve over a finite field. We refer to [11] for more comprehensive description on how these groups, pairing and other parameters are defined.

B. Computational problems

Here, we present some computational hard problems, which will form the basis of security of IBC and HIBC.

Discrete Logarithm Problem (DLP): Given two elements $R, S \in G_1$, find an integer $a \in Z_q^*$, such that $S = aR$ whenever such an integer exists.

Computational Diffie-Hellman Problem (CDHP): For any $a, b \in Z_q^*$, given $< P, aP, bP >$, compute abP.

Decisional Diffie-Hellman Problem (DDHP): For any $a, b, c \in Z_q^*$, given $< P, aP, bP, cP >$, decide whether $c \equiv ab \bmod q$.

Bilinear Diffie-Hellman Problem (BDHP): For any a, b, $c \in Z_q^*$, given $< P, aP, bP, cP >$, compute $e(P, P)^{abc}$.

Gap Diffie-Hellman Problem (GDHP): A class of problems, where DDHP can be solved in polynomial time but no probabilistic polynomial time algorithm exists which can solve CDHP.

3. System architecture

CDS extends HLA with cloud computing technology to realize the service-oriented simulation support environment. Figure 1 shows the system architecture of CDS. Simulation application layer includes various simulation applications. Simulation cloud portal/tool layer provides interaction environment between simulations and users. Simulation model layer provides various simulation models, which are basic elements to construct simulations. The simulation model is the abstract of a kind of simulators and must be deployed to CDS. It is a cloud service and every service instance represents a federate. In this way, simulation models do not need to be placed on the local. Users can invoke them on demand through service access over the wide area network. And multi-users can access different instances of the same model service simultaneously. Services can communicate with each other in spite of the diversity of programming languages and platforms. Cloud infrastructure layer exploits cloud platform [8,9] to integrate and operate various heterogeneous resources, and supports the implementation of core service layer. Heterogeneous resource layer includes various hardware and software resources like computing resources, storage resources, network resources, information resources, and so on. They are not directly exposed to the service layer, but discovered and managed by the cloud platform. Core service layer

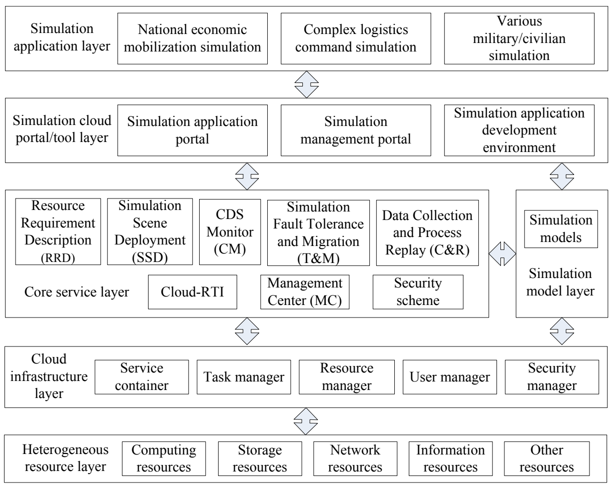
Simulation application layer
National economic mobilization simulation
Complex logistics command simulation
Various military/civilian simulation
Simulation cloud portal/tool layer
Simulation application portal
Simulation management portal
Simulation application development environment
Resource Requirement Description (RRD)
Simulation Scene Deployment (SSD)
CDS Monitor (CM)
Simulation Fault Tolerance and Migration (T&M)
Data Collection and Process Replay (C&R)
Core service layer
Cloud-RTI
Management Center (MC)
Security scheme
Simulation models
Simulation model layer
Cloud infrastructure layer
Service container
Task manager
Resource manager
User manager
Security manager
Heterogeneous resource layer
Computing resources
Storage resources
Network resources
Information resources
Other resources

Figure 1 CDS architecture.

is the most important part. It contains components that collaboratively implement the system functions, based on which CDS can support the entire simulation process effectively. The core components work as follows.

The Cloud-RTI component encapsulates the functions of the traditional RTI as cloud services to support large-scale simulations on CDS. CDS can also support simulations on traditional RTIs by connecting traditional federates to the traditional RTI scheduled by Management Center. Thus CDS has good compatibility, which makes it universal.

Management Center (MC) manages different versions of RTIs and various types of simulation models. It provides registration and query interfaces for them. MC also schedules appropriate RTIs and simulation model services for the registered simulations. Using MC, we can realize share and reuse of simulation resources effectively. The security scheme provides efficient authentication and access control solutions when users want to access simulation model services, thus protecting system security.

The CM component manages the whole simulation federation effectively. The RRD component generates the resource requirement document and the SSD component deploys model services to the appropriate cloud nodes according to the document. The C&R component can record the simulation data automatically and replay the process. The T&M component can implement the fault tolerant mechanism, and migrate federates from overloaded nodes to other appropriate nodes to enhance the load balancing capability of CDS.

4. Key technologies

The key technologies of CDS include the design of Cloud-RTI, MC and the security scheme, which make the simulation run efficiently and safely, and also support other components in the core service layer effectively.

4.1. Cloud-RTI

Cloud-RTI adopts an agent-based invoking and callback strategy and encapsulates the traditional RTI (here we use CRTI [10]) to realize the cloud-based RTI using web service technology. Figure 2 shows the framework of Cloud-RTI. To explain the strategy of Cloud-RTI clearly, we also depict other relevant parts.

In Figure 2, users use the simulation cloud portal to register simulation federation and federates at MC and start simulation. During the simulation, the service-based federates interact with Cloud-RTI effectively over the wide area network. Cloud-RTI consists of four parts, which are HLA invoking interface view, federate request agent, CRTI callback agent and CRTI. HLA invoking interface view provides cloud service interfaces, which implement the six major types of functions of HLA specification. When a federate invokes the service interface of Cloud-RTI, federate request agent delivers the request message to CRTI, invoking the CRTI function. CRTI callback agent receives the callback of CRTI, generates callback events and puts them in the callback queue, which makes CRTI return from callback functions rapidly and improves the performance of Cloud-RTI. Callback event processor executes the events and invokes the corresponding callback service interfaces of the federate. The federate request agent and CRTI callback agent make the service-based federate interact with Cloud-RTI effectively over the wide area network, thus, the whole simulation can run efficiently. The agent mechanism makes Cloud-RTI flexible and can integrate different traditional RTIs into Cloud-RTI for application requirements, which makes CDS practical.

4.2. Management Center

MC is like glue in CDS. It contains resource integration layer and resource scheduling layer. The resource integration layer interacts with simulation portals, various versions of RTIs and simulation model services, accepting the requests of query or registration from these components. Users use simulation portals to register federations and federates at MC and each federate has two different logic views, which are a RTI-level view and an application-level view. The two views describes the states of federates at RTI level and application level respectively. These states are reported to MC by RTI and federates, and each view has three states: waiting, running, exit or finish. The resource scheduling layer works as follows: when a user registers a simulation federation, it refers to the provided parameters and the current performance of RTIs, and then adopts a scheduling mechanism based on the supported

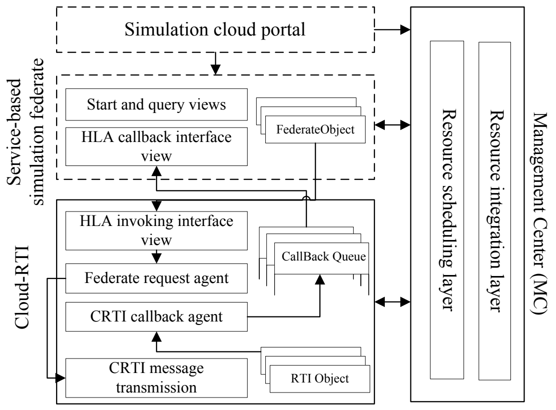
Simulation cloud portal
Service-based simulation federate
Start and query views
HLA callback interface view
FederateObject
Cloud-RTI
HLA invoking interface view
Federate request agent
CallBack Queue
CRTI callback agent
CRTI message transmission
RTI Object
Resource scheduling layer
Resource integration layer
Management Center (MC)

Figure 2 The framework of Cloud-RTI.

federate number (SFN) to schedule an appropriate RTI dynamically for the registered federation, thus achieving dynamic load balance for the system. To implement the simulation, the resource scheduling layer also schedules the required simulation model services for the federation. The SFN-based RTI schedule mechanism is described as follows.

SFN refers to the number of federates which can be still supported by RTI currently. SFN is calculated by the current federate number (CFN) which is supported by RTI now, and the current performance of RTI, which includes memory utilization α, CPU utilization β and network utilization γ. The sum of SFN and CFN is the maximum number of federates which can be supported by RTI.

Assume that when there are no federates running on the RTI, the values of α, β, γ are α_0, β_0, γ_0 respectively. To ensure that the system can run smoothly, the maximum value of α, β, γ are set to $\alpha_{\max}$, $\beta_{\max}$, $\gamma_{\max}$. SFN_α, SFN_β, SFN_γ are the federate numbers which can be still supported by memory, CPU and network currently, which are calculated as follows:

$$SFN_\alpha = \frac{\alpha_{\max} - \alpha}{\alpha - \alpha_0} \times CFN \tag{1}$$

$$SFN_\beta = \frac{\beta_{\max} - \beta}{\beta - \beta_0} \times CFN \tag{2}$$

$$SFN_\gamma = \frac{\gamma_{\max} - \gamma}{\gamma - \gamma_0} \times CFN \tag{3}$$

To ensure that the system can run smoothly, it's required that every resource cannot exceed its maximum. Thus, SFN is calculated by selecting the minimum among SFN_α, SFN_β, SFN_γ, shown as follow:

$$SFN = MIN(SFN_\alpha, SFN_\beta, SFN_\gamma) \tag{4}$$

When the RTI first starts, there are no federates running on it, then $CNF = 0$, $\alpha = \alpha_0$, $\beta = \beta_0$, $\gamma = \gamma_0$, thus the above formulas are not applicable in this case. Suppose that the numbers of federates which can be supported by a unit of each resource are $\varDelta_\alpha$, $\varDelta_\beta$, $\varDelta_\gamma$ respectively. The initial value SFN_0 is calculated as follow:

$$SFN_0 = MIN((\alpha_{\max} - \alpha_0) \times \varDelta_\alpha, (\beta_{\max} - \beta_0) \times \varDelta_\beta, (\gamma_{\max} - \gamma_0) \times \varDelta_\gamma) \quad (5)$$

To achieve dynamic load balance for the system, when MC schedules a RTI for the simulation, it calculates SFNs of all registered RTIs according to their performance which are reported by RTIs periodically, then selects the RTI with the maximum SFN, and allots it to the simulation.

4.3. Simulation interaction flow

Figure 3 shows the interaction flow of a simulation carried out in CDS. Firstly, various simulation model services and RTIs are registered at MC (step 1 and 2). Then a simulation client gets the registered information and registers a simulation federation. MC schedules an appropriate RTI for registered federation (step 3). Next, the client queries MC for the required simulation models and MC schedules the model services if the services agree (step 4). Then the client registers federates at MC. A RTI-level view and an application-level view for each federates are established (step 5). The state of the views is waiting.

After that, the client starts the simulation through the simulation portal. It gets instances from the factories of the model services (step 6), and then starts the instances (step 7). The instances inform MC to refresh the state of the application-level view. The state now is running (step 8). Next, the instance of model service requests Cloud-RTI factory for the instance of RTI ambassador service (step 9), and then initializes the service instance (step 10). The instance of RTI ambassador service informs MC to refresh the RTI-level view. The state now is running too (step 11). The two types of service instances interact with each other. They invoke the service interfaces of the other through the endpoint information of the other to implement the logic of the simulation (step 12).

At the end of the simulation, the instances of model services quit the federation (step 13) and inform MC to refresh the RTI-level view. The state now is exit (step 14). The client destroys federates (step 15). The state of the application-level view is refreshed to finish (step 16). At last, the client informs MC to destroy federation and delete all views of federates (step 17).

4.4. An efficient security scheme

In CDS, various simulation model services, which are large-scale and dynamic, are provided by many service providers which belong to different organization domains. Because each of these organizations has its own identity management and access control system, thus a user who wants to

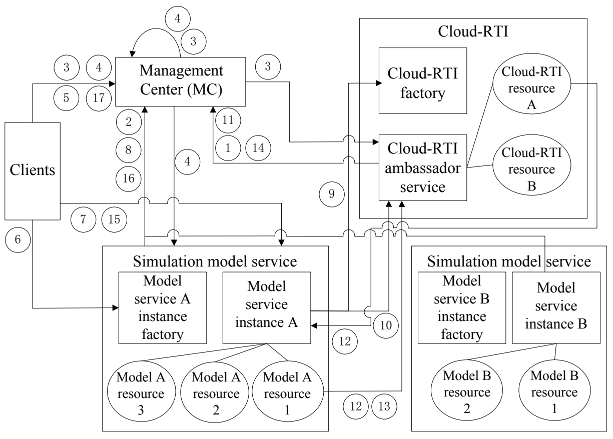

Figure 3 Simulation interaction flow of CDS.

access services from different organizations needs multiple digital identities from these organizations to authenticate itself for each service, which will bring great inconvenience for users. Meanwhile, different users also belong to different organization domains. Providing effective authentications for users when accessing services from different organization domains would be difficult. In the current HLA standard and existing simulation grid systems, there are no efficient security solutions for large-scale distributed simulations. To solve this problem, we propose a security scheme for CDS based on hierarchical identity-based cryptography (HIBC) together with role-based access control (RBAC) [15].

In CDS, there is a root KGC in the overall domain, and each sub-level domain (organization domain) within CDS also has its own KGC. The root KGC manages the whole CDS, each organization domain is the first level and users in these domains are the second level. The root KGC will allocate and authenticate identities for all the organization domains. Each organization domain uses its own domain KGC to allocate and manage the identities and roles of all the users in its domain. Each user in this domain has its own identity and this identity is a hierarchical identity, which includes the identity of the domain, the identity of the user and the identifiers of the user's roles.

We assume there are n domain KGCs. The security scheme consists of the following steps: setup, extraction, signing and verification, which are described as follows:

Setup:

The root KGC:

1. generates groups G_1, G_2 of prime order q and a bilinear map $e : G_1 \times G_1 \to G_2$;
2. chooses an arbitrary generator $P \in G_1$;
3. picks a series of secret master keys $s_1, s_2, \ldots, s_n$ from Z_q^* at random and computes the corresponding public keys $\{Q_i = s_i P | i = 1, 2, \ldots, n\}$;
4. chooses cryptographic hash functions $H_1 : \{0, 1\}^* \to G_1$ and $H_2 : \{0, 1\}^* \to G_1$;

5. publishes the system parameters (G_1, G_2, e, P, $\{Q_1, Q_2, \ldots, Q_n\}$, H_1, H_2).

Extraction:
For the sub-level domain KGC with identity ID_i (abbreviated as KGC$_i$),$1 \leq i \leq n$, the root KGC:

1. computes its public key $P_k = H_1(ID_i)$;
2. computes its private key $s_k = s_i P_k$ and gives s_k to the KGC$_i$.

For a user with identity ID_u in the domain, we assume its roles are (R_1, ..., R_m). KGC$_i$:

1. computes its public key $P_u = H_1(ID_i, ID_u||R_1|| \ldots ||R_m)$;
2. computes its private key $s_u = s_k + s_k P_u$ and gives s_u to the user;
3. computes $Q_k = s_k P$ and gives Q_k to the user.

Signing:
The user with identity ID_u:

1. selects a random message m and computes $h_m = H_2(ID_i, ID_u||R_1|| \ldots ||R_m, m) \in G_1$;
2. computes $Sig_m = s_u + s_u h_m$;
3. sends Sig_m, Q_k and $Q_u = s_u P$ to the service provider.

Verification:
Let (Sig_m, Q_k, Q_u) be the signature for the user with identity ID_u and message m. The verifier confirms that:

$$e(P, Sig_m) = e(Q_i, P_k)e(Q_u, h_m)e(Q_k, P_u)$$

For any service provider in some organization domain, it is easy to authenticate the users that access the services it provides by the proposed security scheme. After that, the service provider executes the authorization for users according to RBAC, which is a general useful method for common resource access of enterprise level systems. It distributes the permissions to each role and one user gets the resource access authorization through its roles. Note that in the security scheme, the root KGC picks a series of secret keys $s_1, s_2, \ldots, s_n$ at random. Each of them is for one sub-level domain KGC respectively. The method can enhance the system security. If one secret key for a domain KGC is disclosed accidentally, other domains are still secure and the key refreshment will be done just in this domain. The security of the scheme is based on the difficulty of the computational hard problems presented in section 2.3, and the security proof can be derived from that of the hierarchical ID-based signature scheme in [12], which is straightforward. Thus we omit the proof. Because of the traits of HIBC and RBAC, the security scheme is efficient and scalable.

5. Performance evaluation

We developed CDS using Java and a practical simulation application, which is a sushi restaurant simulation [10], based on CDS to effectively evaluate the performance of Cloud-RTI and MC. In our evaluation, we use one MC server, seven RTI server, one simulation portal and five clients to construct the simulation. The experimental environment is as follows: Intel CoreTM 2 Duo 1.83 GHz CPU, 2 GB memory, 100Mbps bandwidth and Linux system.

We developed the simulation federation based on Cloud-RTI and traditional CRTI respectively. The federation simulates the process of sushis production, transportation and consumption. It has five federates, which are manager, production, transport, consumption and viewer. Each federate is both time-constrained and time-regulating. During simulation, the federation proceeds up to one thousand time advances, and the time interval between advances is one second. During every advance, when a federate finishes its task, it waits and performs the next advance when one second interval arrives. During each advance, the longer the waiting time (or idle time) is, the better the performance is. Figure 4 represents the idle time of the federation during each advance on Cloud-RTI and CRTI. Table 1 compares their idle time in details. It can be seen that the peak performance of Cloud-RTI and CRTI is approximate, while the overall performance of Cloud-RTI declines slightly and the stability of Cloud-RTI needs to be improved further. The reason is that cloud-based federates communicate with Cloud-RTI through SOAP and HTTP protocol, and data packets need extra packing and unpacking processes of protocol stacks before they are passed to the encapsulated RTI. In our future work, we will adopt faster SOAP message transmission to reduce the performance gap between Cloud-RTI and the traditional RTI.

MC can complete operations in milliseconds for all the requests in our evaluation, as shown in Table 2. The time of federation registration is the longest, since during federation registration, an appropriate RTI is scheduled for the federation using the proposed RTI schedule mechanism. This operation involves the interaction of multiple services. The time of federate scheduling is longer than the rest operation time, because the proposed security scheme must be enforced during federate scheduling process, which will cause extra time. However, the extra time is just tens of milliseconds, which has little impact on the efficiency of the simulation. For the rest operations, MC can finish them in the local rapidly. A resource discovery system was proposed in [16] for grid-based distributed simulation, and there is also a MC in this system. However, the MC only supports the registration of traditional RTIs. The RTI schedule mechanism in their scheme is also different from our scheme. For example, in their scheme, when retrieving the information of RTIs, MC needs to interact with two services immediately and the current efficiency of RTIs are not considered. In our scheme, MC makes optimization in the process of registration operations. When a simulation federation is registered, an effective RTI will be allotted to it immediately. In their scheme, the time spent in federation registration and information retrieval is up to 10 s

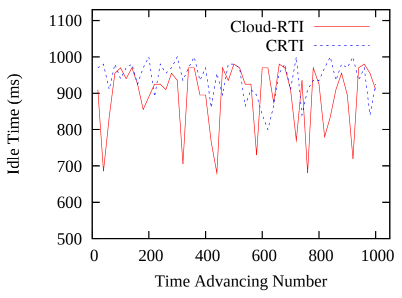

Figure 4 The idle time of the federation during advances.

Table 1 The comparison of idle time statistics (ms).

RTI	Mean	Maximum	Minimum	Fluctuation	Variance
Cloud-RTI	905.1	985	690	295	75.1
CRTI	950.2	998	801	197	47.3

Table 2 The operation time of MC (ms).

Operations	Time
Federation registration	328
Federation list retrieval	47
Federate scheduling	100
Federation details retrieval	46
RTI list retrieval	47

totally. Besides, security issues are not considered in their scheme.

6. Conclusion

Cloud computing technology can enhance the capability of HLA and provide an effective solution for large-scale distributed simulation systems. In this paper, we propose an efficient and secure cloud-based distributed simulation system (CDS) and describe its key technologies in details. It provides a service-oriented simulation support environment, which has some significant advantages for large-scale distributed simulations. For example, CDS can provide users with on-demand simulation resources, run simulations on the wide area network efficiently, realize share and reuse of simulation resources, improve load balancing capability of the simulation, and provide security guarantee. Compared with the previous simulation grid systems, CDS is efficient, secure, universal and practical. In the future work, we will focus on more effective RTI schedule mechanism and further improving the efficiency of Cloud-RTI.

Acknowledgement

This work is supported by the National Natural Science Foundation of China with grants 61173170 and 60873225, the National High Technology Research and Development Program of China (863 Program) with grant 2007AA01Z403.

References

[1] IEEE Std 1516-2000, 1516.1-2000, 1516.2-2000, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)(2000).
[2] B. H. Li, X. D. Chai, B. C. Hou, T. Li, Y. B. Zhang, H. Y. Yu, J. Han, Y. Q. Di, J. J. Huang, C. F. Song, Z. Tang, P. Wang, G. Q. Shi and X. H. Wang, Networked modeling & simulation platform based on concept of cloud computing—cloud simulation platform, J. Syst. Simulat. **21**, 5292-5299(2009).
[3] C. F. Song, B. H. Li, X. D. Chai, T. Zhen and D. Xi, Research on developing modes of simulation system in simulation grid, In Proc. of the 12th International Conference on Computer Supported Cooperative Work in Design, IEEE Computer Society, Los Alamitos, 535-541(2008).
[4] K. Zaajac, A. Tirado-ramos, Z. Zhao, P. Sloot and M. Bubak, Grid services for HLA-based distributed simulation frameworks, In Proc. of the 1st European Across Grids Conference, Springer-Verlag, Heidelberg, 147-154(2003).
[5] D. Chen, G. Theodoropoulos, S. Turner, W. Cai, R. Minson and Y. Zhang, Large scale agent-based simulation on the grid, Future Gener. Comp. Sy. **24**, 658-671(2008).
[6] M. Games, Federation grid, http://www.federationgrid.org (2004).
[7] H. He, L. Chen, P. P. Yuan, X. Xu and X. F. Wang, A security architecture for grid-based distributed simulation platform, In Proc. of the IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, IEEE Computer Society, Los Alamitos, 207-212(2008).
[8] R. Buyya, Cloud computing: The next revolution in information technology, In Proc. of the 1st International Conference on Parallel Distributed and Grid Computing, IEEE Computer Society, Los Alamitos, 2-3(2010).
[9] K. Chen and W. M. Zheng, Cloud computing: System instances and current research, J. Softw. **20**, 1337-1348(2009).
[10] F. Kuhl, J. Dahmann and R. Weatherly, Creating computer simulation systems: An introduction to High Level Architecture, Prentice Hall PTR, Upper Saddle River, NJ(2000).
[11] D. Boneh and M. Franklin, Identity-based encryption from the Weil pairing, In Proc. of the 21st Annual International Cryptology Conference on Advances in Cryptology, Springer-Verlag, Heidelberg, 213-229(2001).
[12] C. Gentry and A. Silverberg, Hierarchical ID-based cryptography, In Proc. of the 8th International Conference on the Theory and Application of Cryptology and Information Security, Springer-Verlag, Heidelberg, 548-566(2002).
[13] H. W. Lim and K. G. Paterson, Identity-based cryptography for grid security, In Proc. of the 1st IEEE International Conference on e-Science and Grid Computing, IEEE Computer Society, Los Alamitos, 394-404(2005).
[14] L. Yan, C. M. Rong and G. S. Zhao, Strengthen cloud computing security with federal identity management using hierarchical identity-based cryptography, In Proc. of the 1st

International Conference on Cloud Computing, Springer-Verlag, Heidelberg, 167-177(2009).

[15] R. S. Sandhu, E. J. Coyne, H. L. Feinstein and C. E. Youman, Role-based access control models, IEEE Computer **29**, 38-47(1996).

[16] W. B. Zong, Y. Wang, W. T. Cai and S. J. Turner, Grid services and service discovery for HLA-based distributed simulation, In Proc. of the 8th IEEE/ACM Distributed Simulation on Real Time Application Symposium, IEEE Computer Society, Los Alamitos, 116-124(2004).

Heng He received his M.S. degree from School of Computer Science and Technology at Huazhong University of Science and Technology in 2007. Now he is a Ph.D. candidate in the Intelligent and Distributed Computing Laboratory, School of Computer Science and Technology, Huazhong University of Science and Technology. His research interests include cloud computing, distributed simulation, and network security.

Ruixuan Li received the B.S., M.S., and Ph.D. degrees from School of Computer Science and Technology at Huazhong University of Science and Technology in 1997, 2000, and 2004 respectively. He is currently a professor of School of Computer Science and Technology at Huazhong University of Science and Technology, and is the director of the Intelligent and Distributed Computing Laboratory. His research interests include distributed system security, cloud computing, information retrieval, peer-to-peer computing, and social network.

Xinhua Dong received his M.S. degree from School of Computer Science and Technology at Huazhong University of Science and Technology in 2008. Now he is a Ph.D. candidate in the Intelligent and Distributed Computing Laboratory, School of Computer Science and Technology, Huazhong University of Science and Technology. His research interests include cloud computing, information retrieval, and big data management.

Zhi Zhang received his M.S. degree from School of Computer Science and Technology at Huazhong University of Science and Technology in 2005. Now he is a associate professor of School of Computer Science and Technology at Wuhan University of Science and Technology. His research interests include semantic web, cloud computing and social network.

Hongmu Han received his M.S. degree from School of Computer Science and Engineering at Wuhan Institute of Technology in 2007. Now he is a Ph.D. candidate in the Intelligent and Distributed Computing Laboratory, School of Computer Science and Technology, Huazhong University of Science and Technology. His research interests include cloud security, and access control.