Future Generation Computer Systems 86 (2018) 185–198

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Software architecture and algorithm for reliable RPC for geo-distributed mobile computing systems

Asmat Ullah Khan, Susmit Bagchi *

Department of Aerospace and Software Engineering (Informatics), Gyeongsang National University, Jinju, Republic of Korea

HIGHLIGHTS

- GMCC-RPC: Reliable mobile RPC for geo-distributed systems.
- Software architecture for mobile and reliable RPC for geo-distributed systems.
- Mobile and Reliable RPC using server chains in geo-distributed systems.

ARTICLE INFO

Article history:
Received 6 January 2018
Received in revised form 3 April 2018
Accepted 7 April 2018
Available online 18 April 2018

Keywords:
Remote procedure call
Distributed systems
Interprocess communication
Cloud computing
Mobile computing
Geo-distributed mobile cloud computing
Smart mobile devices

ABSTRACT

Remote Procedure Call (RPC) is a computing as well as communication model for distributed processes to execute client routines on remote servers in the distributed systems. Due to the evolution of geo-distributed mobile cloud computing systems, mobile devices are exposed to frequent disconnection due to limited battery lifetime, processing capacity and network bandwidth while roaming globally. The existing standard RPC and mobile RPC frameworks are not completely suitable for applications in geo-distributed mobile cloud computing. This paper proposes a novel software architecture and associated algorithms for realizing reliable RPC under global mobility of clients. The stateful server chaining and multiple authentication primitives are employed in the proposed design to achieve security as well as location transparency. The software architecture is implemented on heterogeneous testbed and evaluated with promising results. The heterogeneity of mobile cloud platform is considered in the design by employing specific XDR format enhancing portability. A detailed comparative analysis of the proposed design is included in the paper.

1. Introduction

The basic concept of Remote Procedure Call (RPC) is originated from the model of data movement during local procedure call as an extension to networked systems. Unlike traditional network applications, where communications are often based on low-level message passing primitives offered by the transport layer, the distributed systems need powerful and flexible facilities for communication between processes. RPC is an Interprocess communication (IPC) mechanism used for building distributed applications. RPC mechanism follows client/server (*C/S*) model for execution of local calls on the remote servers exploiting remote resources [1–4]. The applications of RPC can be found in distributed systems such as Amoeba distributed operating system, Sprite network operating system and Andrew File System [5]. Furthermore, RPC is a mechanism for providing synchronized type-safe communication between two processes [1,6].

Libraries executing RPC methods can be found in different applications like Facebook Thrift or web services with Google Protocol Buffer or in the domains such as grid computing (GridRPC) [2]. These standard RPC skeletons on high-performance computing (HPC) systems have two major constraints. First, they are unable to take advantage of local transport procedure in order to transfer data efficiently because these frameworks are based on the TCP/IP protocols. Second, they are unable to transfer bulk data due to the limitation imposed by RPC interface (limited to megabyte level) [2]. However, shipping bulk data through RPC library is not a recommended solution due to overhead arising from serialization, encoding and copying data multiple times before reaching to a remote destination [2]. Inter-process byte stream is more effective approach as compared to RPC to transfer unstructured bulk data [7].

Although RPC is used as a foundational unit for distributed systems applications, yet it is not adequate for supporting large-scale distributed systems used in datacenters [8]. Unlike traditional technologies where resources of single cloud provider or datacenter were used, todays applications are focused to exploit cloud

* Corresponding author.
E-mail address: profsbagchi@gmail.com (S. Bagchi).

https://doi.org/10.1016/j.future.2018.04.023
0167-739X/

infrastructure by using heterogeneous resources of various cloud providers involving mobile clients [9]. The degree of complexity is increased in remote execution as it contains issues such as, establishing a connection with the server, marshaling input data, construction of message structures, interacting across the network and, resolving potential error cases [10].

With the evolution of computer technology, new paradigms are opened in the form of mobility of nodes and enhanced computation speed due to enhanced wireless network bandwidth. As a result, the concept of mobile cloud computing (MCC) is introduced [11–14]. Modern users demand advanced services and computing power on the mobile devices [3,15]. A wide range of mobile applications like emailing, chatting, internet browsing and gaming are introduced in smartphones and tablets [16]. These Smart Mobile Devices (SMDs) are proficient towards computation using offloading techniques [13,17]. These devices initiate RPC to offload and execute computation intensive programs on remote systems in the cloud using annotation, special compilation or binary modification [13]. This approach has two limitations. First, the access to a cloud provider and configuring their resources using API are not fully matured and standardized [18]. Second, the reduction of data transfer is required as network bandwidth is the most crucial resource in mobile cloud systems [18,19]. The mobility management of nodes imposes further challenges. This paper aims to address these issues in designing reliable RPC framework in globally mobile RPC paradigm.

1.1. Motivation

The primary aim of RPC mechanism is to give a type-safe procedure to application programmers of distributed systems [20]. The key challenges in building mobile distributed RPC are: (a) how to couple RPC messages with a network resource and make this integration executable in user address space [8], (b) how to ensure the reliability of RPC execution in a distributed environment where the system experiences differing degrees of connectivity between nodes [21], (c) how to handle delay (locally or remotely) in such environment, (d) how to define the semantics of a call when an error occurs due to a node or wireless connection failure, (e) how to integrate these remote calls into mobile computing systems and, (f) how to ensure the integrity, portability and, security of data over the network [6]. The mobile RPC architecture should be able to resolve messaging overhead with the increasing number of distributions of mobile clients. Generally, a mobile client must be efficient to locate remote resources before accessing them in order to reduce degree of distribution (localities) in distributed systems. However, mobile applications are exposed to technical constraints in a wireless network. Hence, mobile devices face difficulties to locate remote resources in large-scale geo-distributed systems. Due to the power and bandwidth limitations as well as high communication errors in wireless network, the traditional RPC frameworks are not suitable to execute mobile RPC [3]. The design constraints and requirements are further enhanced in geo-distributed large-scale distributed systems involving mobile clients. It is required that, the number of messages exchanged between a server and a mobile client be optimized. As the degree of distribution is increased, locating and accessing a resource in geo-distribution become more challenging. The system must have an efficient mechanism to locate and access a dynamic resource [10]. Traditional RPC frameworks consider that all hops in communication networks are stationary, predictable and accessible. These designs do not count bandwidth as a scarce resource. Therefore the present RPC architecture does not support mobility of clients connected by wireless network resulting in failure in execution and showing inconvenience in case of disconnection [21,22]. M-RPC [22] enables dynamic binding, disconnected operations and call retrieval from mobility support router (MSR). Coda [22] deals with disconnected operations by integrating stand-alone client file system with file server after reconnecting at application layer. However, this approach cannot be used as a generalized solution to deal with all kinds of mobile applications as each application will demand separate approaches to detect and resolve disconnection. Researchers have proposed RPC chaining mechanism, which enables applications to reduce performance degradation at enterprise level using wide area network [23]. However, it does not consider global mobility of clients. GPUrpc [17] design is aimed at graphical processing unit used for high performance computing systems utilizing computation intensive applications involving environmental recognition such as, augmented reality (AR). However, devices having low computational power do not validate these computing essentials. Furthermore, researchers have proposed process state synchronization mechanism in mobile cloud systems, which is implemented as kernel module in order to achieve high performance [14].

This paper proposes a model of mobile RPC software architecture intended for geo-distributed mobile computing paradigm. The proposed architecture is focused in providing variable services to mobile clients in GMCC (Geo-distributed Mobile Cloud Computing) systems. The architecture enables dynamic adaptation of mobile devices to changing environments. In the proposed model, the disconnectivity problem of the mobile clients is handled by the chained stateful servers having data IO with specifically designed portable XDR format. In our approach, RPC of a mobile client is supported through a chain of secondary servers, which provide chain service to a mobile client. All authentications regarding a mobile client are performed on both servers (primary and secondary) in order to achieve maximum security. In case of call failure due to packet loss on wireless network or server busy–wait state, we have added re-transmission facility in our design. Disconnectivity is resolved by keeping both servers stateful to a limited time interval. The distinctive characteristics of our design are as follows:

- Mobility Support to clients by server chaining during RPC execution in geo-distributed mobile computing systems.
- High reliability of RPC under global mobility of clients.
- Efficient *C/S* and *S/S* synchronization models with multiple authentications.
- Specifically designed XML XDR for portable data retrieval over the network for uniformity of execution in a heterogeneous environment.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 explains the architectural design and, Section 4 presents algorithm design. Section 5 describes implementation of proposed architecture. Section 6 highlights performance evaluation. Section 7 presents a detailed comparative analysis of our model with various RPC frameworks and, finally Section 8 concludes the paper.

2. Related work

Distributed systems rely on several computing models and respective implementations. One of the popular model and implementation is the Open Software Foundation's Distributed Computing Environment (DCE) [24]. RPC mechanism spans application layer and transport layer in Open System Interconnection model of network communication. It is an easy way to develop an application that contains multiple processes distributed over the network. One of the key features of RPC is that it is used as a communication protocol at ISO Presentation Layer [25]. Athena RPC is a project developed at MIT to incorporate different communication resources for academic use [5]. It is built under tight constraints such as, avoiding adaptations to Unix kernel, assisting