International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 07, July-2021

Crud Service for Client and Server Timing Computation using HTTP Methods

Manish L. Jivtode
Department of Computer Science,
Janata Mahavidyalaya, Chandrapur, Maharashtra, India

***Abstract*:- With the increasing popularity of the cloud, web service technology has emerged as a popular way for building distributed applications involving distributed databases. The sensitive data exchanged between web services and their clients hosted in the cloud data centers must be protected while in the transit. The distributed databases and applications in these geographically dispersed data centers belong to either internet or intranet domains.**

In this paper, study of weakness and suggest improvements of CRUD service contract and a set of operations like CREATE, RETRIEVE, UPDATE, and DELETE (CRUD) by using HTTP verbs like POST, GET.

Keywords: Web Service, Distributed database, REST web Services, CRUD service, GET method and POST method.

1. INTRODUCTION

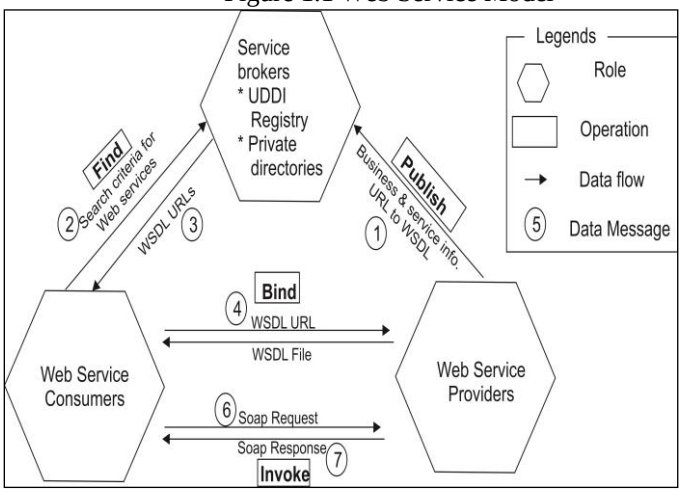
Recent improvement of web services technology provide uninterrupted, lightweight, continuous web services to resource constrained devices like mobile, laptop, desktop, tablet etc. in wireless or wired environment. The services are based on service oriented principles and follows four tenets of service oriented development

(i) Boundaries are explicit,
(ii) Services are autonomous,
(iii) Services share schema and contract, not class,
(iv) Service compatibility is determined based on policy.

Web services propose a service-oriented paradigm for computing in which distributed, loosely coupled services collaboratively implement business processes and accessed via the Internet by end users. The use of Web services over the intranet and internet has increased rapidly. Web services are used to support application-to-application communication and to address interoperability issue for systems integration project. These Web services provide a standard-based approach for different software applications or components involved in supporting real-time information retrieval or presenting dynamic context-driven information to the user. Web services rely on a set of standards to support interoperability among applications developed in different languages and running on different platforms or operating systems.

Core web services standards including SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language), and UDDI (Universal Description, Discovery, and Integration).

Figure 1.1 Web Service Model



(Source: Borrowed from [2])

Web Services are considered an instance of an SOA. The web services network is an application level network involving a number of participants- providers, service consumers, and service registry operators.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 07, July-2021

2. REPRESENTATIONAL STATE TRANSFER (REST)

REST is architecture for developing web services. REST architecture uses HTTP or similar protocols, by constraining the interface to a set of well-known standard operations (i.e., GET, PUT, POST, and DELETE for HTTP). REST architecture is designed to show how existing HTTP is enough to build a Web service and to show its scalability [1]. Roy Fielding, who is also one of the principal authors of HTTP, in his doctoral thesis [7], "Architectural Styles and the Design of Network-Based Software Architectures," defined a set of architectural principles for the Web called REST. REST architectural principles can be used for designing Web service solutions. REST focuses on a system's resources, including how resource states are addressed and transferred over HTTP [4]. The main idea behind REST was to use well-developed HTTP for transferring data between machines, rather than using a protocol that works on top of the HTTP layer for message transfers. An application designed following REST principles would use HTTP to make calls between the machines, rather than relying on complex mechanisms like CORBA (Common Object Request Broker Architecture), RPC (Remote Procedure Call), or SOAP. Therefore, REST applications use HTTP request functions to post data, read data, and delete data, thus using the full functionality of HTTP Create, Read, Update, and Delete (CRUD) operations. REST can run on HTTPS, providing for the secured transmission of data. The CRUD operations in conjunction with HTTP REST functions are shown –

Table 1.1 HTTP methods and CRUD action

CRUD operation	REST keywords (HTTP)
READ-read, retrieve	GET
CREATE-create or add new entries	POST
UPDATE-update or edit existing data	PUT
DELETE-delete existing data	DELETE

(Source: Borrowed from [6])

The application of REST principles for web services requires HTTP protocols find it easy to understand and apply REST principles. The term "RESTful" is like "object-oriented," a language, a framework, or an application that may be designed in an object-oriented way, but that does not make its architecture the object-oriented architecture [5]. RESTful Web services are emerging as the choice for many of the leading Internet companies to expose their internal data and functionalities as URI identified resources.

3. EXPERIMENTAL WORK

[A] DATA SET USED

The availability of the standard data set in the field of security in distributed database is limited, six of the standard dataset collections were considered for the study and performance evaluation. The basis for selection of these datasets is availability for query and relevance judgments with these datasets. These collections are described as follows:

Table 1.2 Data sets used for performance evaluation

Name	Description	Data Size	Number of queries tested			
			GET	POST	PUT	DELETE
Text	Plain text	1KB – 16KB	10	10	10	10
Photo	JPEG Image	1KB – 16KB	10	10	10	10
Audio	MP3	1KB – 16KB	10	10	10	10
Video	WMV	1KB – 16KB	10	10	10	10
Headers	Custom headers	Few bytes	10	10	10	10
Message	Message body	1KB – 16KB	10	10	10	10

(Source: Compiled by Researcher)

Four standard REST web service methods were tested for analyzing performance of the implemented system. They include GET, and POST methods. GET method does not carry payload in the message body in request part and returns data in the response only. The data is enclosed in XML tags and placed in message body of POST requests.

In GET request, an unencrypted plain text request along with custom header (for authentication) is sent to the REST web service and HTTP message body does not contain any text/image/audio/video data. Upon reaching HTTP GET message at REST web service, headers are processed and appropriate resource is returned in encrypted form during GET response. The received data is decrypted at the client side and are verified.

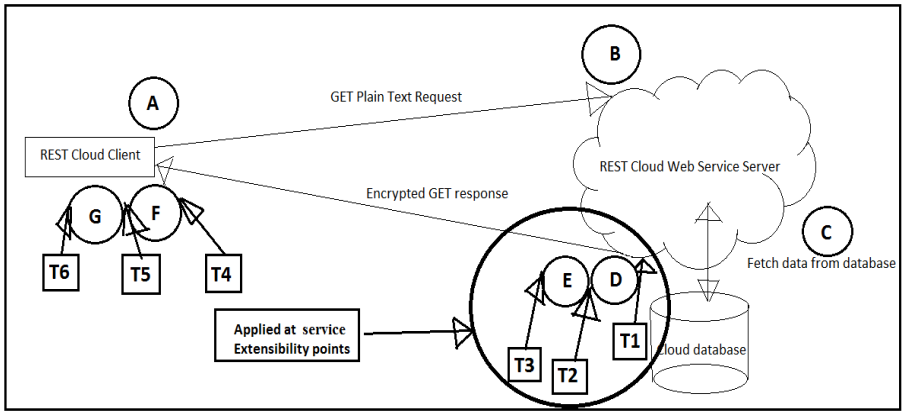
In POST requests, data is signed and then encrypted before sending it through the POST request message body. Signed values of data are asymmetrically encrypted and symmetric key sent through custom headers. In HTTP, message reaches REST service, it is first decrypted using custom header data and then verified by applying encryption/decryption algorithm at the REST web service extensibility points. The messages processed by REST web service by creating a new resource on the REST service database and signed and encrypted response sent back to the REST client. Upon receiving signed and encrypted response at the REST client, it is decrypted and verified and finally presented at the client side. Image/audio/video data is converted to Base64 binary or Base64 string before sending to the REST web service.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 07, July-2021

[B] CLIENT AND SERVER REQUEST / RESPONSE TIMING COMPUTATIONS

GET request sent from REST client, it does not carry any payload and only plain text request sent to the REST service. In GET, client authentication information sent through custom headers. It returns an encrypted response from REST web service. Both REST web service and REST clients are programmed to compute the encryption, decryption timing. GET response return text, JPEG image, audio or video data from REST web service to client. Request timings are calculated by computing time elapsed between point A and point B. GET request may result in retrieving a resource from the local database on the client side.

Figure 1.2 Request/Response Timing for Client and Server



The experiments were performed to record signing, encryption, decryption, verification, request and response timing for various datasets by programming WCF REST web service extensibility points.

GET request/response timings are calculated by computing time elapsed between point A and point B. GET request/response results in retrieving a resource from the local database.

Points A – B -> GET request
Time T1 = DateTime.Now()
//Code for sending request
Time T2 = DateTime.Now()
Request timing during GET = T2 – T1
Points D, E, F, G -> GET response
Response time = (T2 – T1) + (T3 – T2) + Time from E to F +(T5 –T4)
+ (T6 – T5)

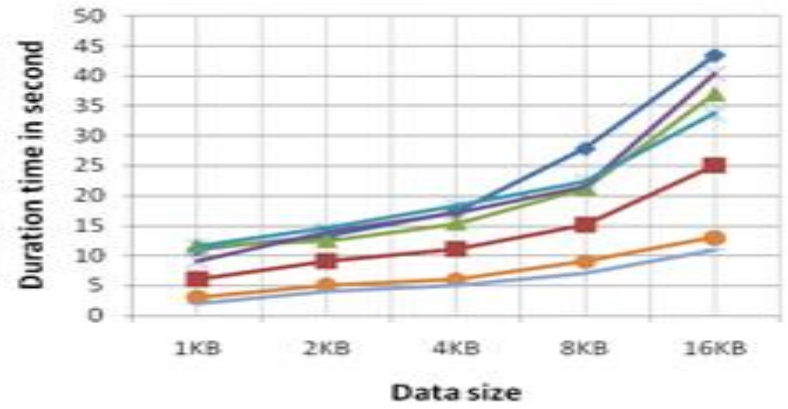
Table 1.3 Client and Serve Request/ Response Time

File size	Request time in milliseconds					Response time in milliseconds				
	Client Side			Server Side		Server Side		Client Side		
1KB	330	-	-	-	-	203	54	1232	90	59
2KB	380	-	-	-	-	255	71	1249	109	67
4KB	394	-	-	-	-	329	98	1278	114	86
8KB	445	-	-	-	-	439	113	1297	169	124
16KB	586	-	-	-	-	474	167	1311	187	199

(Source: Compiled by Researcher)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 07, July-2021

Request / Response Time for Clint and Server in milliseconds



This graph shows the performance measurement of encryption/decryption of various data sizes versus time and clearly indicates that as the data size increases from 1KB to 16KB, the encryption and decryption time also increases.

4. CONCLUSION

From the study, it is concluded that, the size of data is directly proportional to the time that is Fs α Te and Fs α Td, where Fs is the file size. It indicates that δFs = k1δTe, then k1= δFs / δTe and δFs = k2δTd, then k2 = δFs / δTd, where k1 and k2 are the proportionality constant indicates that the rate of change of request time and response time with respect to size of data transferred. GET method scores higher performance than others. Text data set shows best performance compared to image, audio and video data sets since it is uncompressed plain text in original form.

5. REFERENCES

[1] A. Pironti, N. Mavrogiannopoulos."Length Hiding padding for the transport layer security protocol". Internet draft-pironti-tls-length-hinding-00, IETE Secretariat, February 2013.
[2] A. Barth, C. Jackson, and J. C. Mitchell. “Robust defenses for cross-site request forgery”. In CCS’08: Proceedings of the 15th ACM conference on Computer and communications security, page no. 75-88, 2012.
[3] Dropbox model. "REST Application Programming Interface". 2012.
[4] Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. “WSDL 1.1 Document”. March 7, 2011.
[5] A. Langley. Unfortunate current practices for HTTP over TLS, 2011.
[6] A. Sotirakopoulos, K. Hawkey, and K. Beznosov. On the challenges in usable security lab studies: lessons learned from replicating a study on SSL warnings. In Proc. of SOUPS, page no 3-18, 2011.
[7] F. Lascelles. "RESTful web services and signatures". October 2010.
[8] Aberdeen Group, Federated Identity Systems An Executive White Paper, Aberdeen Group Inc., Boston, page no 11, 2007.