BUILDING SECURED XML REAL-TIME INTERACTIVE DATA EXCHANGE ARCHITECTURE - A COMPARISON TEST BETWEEN TCP VS. UDP PROTOCOLS PERFORMANCE

Y. E. Rabadi, J. Lu
[1] University of Huddersfield, Queensgate, Huddersfield HD1 3DH, UK
[2] University of Leeds, Leeds LS2 9JT, UK

ABSTRACT

This research seeks to introduce architecture of new approach of exchanging XML data files between different Services. The importance of this study is to set new protocol standards of interchanging XML data files between different nodes and types of businesses. The main objective and goal of this study is to transmit XML data file in a secured manner coupled with reliability, quality of communication, independent, scalable and flexibility. Therefore, this architecture designed to minimize the risk of any alteration, data loss, data abuse, data misuse of an XML critical business data information been exchanged. As cloud computing, using existing cloud network infrastructure to get advantage of the scalability, operational efficiency, and control of data flow are big consideration in this architecture. A test has been made to measure the performance of RIDX, one by using TCP protocol, and one by UDP protocol. As a result, starting from 4 nodes up to 8 nodes in the cloud, RIDX architecture performance of UDP showed better performance than TCP, but using TCP assures the reliability and lossless of data transmission to all nodes, in addition, RIDX is reliable multicast transmission.

Keywords XML Security, Cloud Computing, XML Architecture, (TCP, UDP, MultiCast) Protocols

1. INTRODUCTION

When you think of how much demand of the IT needs became, your attention comes to focus on how important is Cloud computing turns to be. Cloud computing become a way to increase capacity and add capabilities. Cloud computing as a definition is a term for anything that involves delivering hosted services over the Internet. These services are generally alienated into three categories: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS).

Exchanging data between different organizations and businesses plays an important and essential role of doing businesses, especially if you think of it in terms of cloud computing services providing infrastructure, platform and software as a service. As XML increasingly become a standard format for transmitting data on networks between different businesses, [1, 2, 3], the need of finding secured and efficient techniques of transmitting XML data files is a necessity matter.

Real-Time Interactive Data Exchange aims to place standards of exchanging XML data information electronically between businesses, organizations, and other groups. RIDX (Real-time interactive data exchange) goal in this study is to build a clear infrastructure required for managing XML data real-time interactive exchange; this will need functionality available in such a system, such as

- Tracking the data from origin to destination, scheduling data transfer in case of any communication drop
- XML standards adaptation
- Integration of other services to facilitate the data transition
- Confirmation between all parties of receiving correct data
- Plays as a routing server to route data to the targeted destination
- Recognition and assurance owners of data
- Registering all system activities into data management system, in addition, includes integration of the data between other systems, applications, services or interfaces.

Furthermore, this approach is more concerned on exchanging sensitive information between different parties, which usually the XML data file size is more predictable and has implicitly upper limit of file size; therefore, this approach at this point does not solve exchanging large XML files, large binary files attached to XML (Audio, Video …etc) nor handles any compression techniques processed on the XML file.

2. RELATED WORK

One of the protocols and services has been used to transmit XML data format is SOAP (Simple Object Access Protocol). Evaluation studies of SOAP XML transmission protocol has been made, specifically of the SOAP and XML performance, several of them concluded that the performance of SOAP and XML acquire considerable price when comparing to binary transmission protocols. [6, 7, 8]
An evaluation experiment was conducted about SOAP implementation latency performance, comparing with CORBA/IIOP and Java RMI protocols. As a conclusion, SOAP is enormously slower, therefore, SOAP XML messages not appropriate of transmitting bulk data. [7] Another comparison has been made while SOAP did fare poorly when compared to both binary CDR and the established industry protocol FIX.
Another implementation of exchanging messages is FIX protocol, The Financial Information exchange; which is a messaging standard developed specifically for the real-time electronic exchange of securities transactions and stock markets. [5]. FIX messages consist of tag-value pairs separated by a special delimiter character (SOH, which is ASCII value 0x01), and types of values include strings, integers, floating point values, it is fast and reliable protocol guarantees data delivery. [5]. But this approach does not comply with XML structure, and FIX protocol is purely for financial trades, stock markets and securities trades, and the purpose of it is not transmitting XML data files between different types of businesses.
Another implementation which is derived from FIX protocol, is FpML and FIXML protocols, it is used also for pure financial interchange trading system [9], and does not solve the XML data file transmission between different applications or different types of businesses, furthermore, depends on a predefined DTD structure of a message including tags and values [9], which means, it is not designed for transmitting XML data files as an independent middleware interchange between other types of businesses.
Another approach which is REST (Representational State Transfer), it is a Web Services depend on method of Request-Respond over HTTP protocol, methods such as (PUT, GET, POST…etc), transfers data in a form of XML document. [10]. Therefore, a limitation of over (4 KB) of data makes GET versus POST impossible, and also impossible to encode such data of URI, which gives an error “HTTP Code 414 - Request-URI too long” [11].

3. THE RIDX SYSTEM COMPONENTS

• RIDX COMMUNICATOR: which is the main engine that controls the communication between different systems, from initiation step until receiving final acknowledgment of the destination server, this protocol is an electronic communication of shake hands messages; Sessions is layered on TCP (transmission control protocol). The value is a type of XML text, the value can hold binary values, which means that the text can be encrypted, and by standard, the length of the field is saved at the beginning of the field value itself. Always the last field of any message is for checksum validation of the whole message, the full message contains three major parts, and the body which is contains the XML message, the head and the tail which contains communication information details about the message specification.
• SHREDDER-SPLITTER / JOINER MODULE: this module is responsible of splitting the file into a set of files (chunks) based on criteria which can be customized by the initiator, Perform fast checks (size, offset, CRC32, MD5) in order to detect file corruption and to give the assurance that your files are successfully restored by the receiving server. When a piece is corrupted, message will notify the originator about it, so you just need to get a new copy of that piece, not the whole set. Each chunk encapsulated in the message protocol, this message can be encrypted and translate it to a binary representation, the length of the data is calculated and attached to the value of this data portion. Each shredded piece of file will be sent in a different path to different receivers on the net; that means thru other RIDX routers signaled by online heart beat signal, until it reaches the destination, then the module will recombine (join) them again to the original XML file format. The RIDX listener is responsible of checking the validity of the record (chunk) of data, and this module is responsible of checking the validity of the XML file all in all.
• FORWARDER AND ROUTING ENGINE: This module is responsible for rerouting any message which not meant to be received by the current RIDX system, and acts like a router of the message, to be routed to the final destination. No processing to the data will be made in this routing module, except of reading the header part of the message to distinguish the destination and confirm the source of receiving message, and to register the routing transaction into the data management system. A part of it also checks the checksum of the message to make sure that the message has been received till this point in a pure shape as of the original message. Figure 1.0 illustrates the architecture of the general network structure between different RIDX systems

4. GENERAL WORK PROCESS OF RIDX

• **INITIATION OF PROCESS**: When requesting to transfer XML data initiated by data management system or by any other interfaces integrated to the RIDX system, and after knowing the destination target for the

XML file to be delivered, the Shredder/joiner module will parse the XML file and shred it into several chunks, each chunk will be read and encapsulated in a message, this message contain all the information needed to reach to the destination target. All chunks will be read until all data in XML file is processed. Meanwhile, the ready messages processed in the system will be prepared to be sent, therefore, the system will send a heart beat to the destination target to make sure that the server is up and ready to receive the data transmission. Also a list of potential servers will be prepared and listed from the data management system, the system will send a heart beat to all potential systems to check each server which has been chosen as a candidate routing server is up and ready to receive data transmission. Servers other than the destination target will act as a routing server. All servers that receives heart beat should respond, or else will consider this server is down and will be eliminated from the potential candidates and from the list.

After filtering the servers and check the readiness of all parties to receive data, the RIDX communicator start sending the prepared messages passed from the shredder module to the different parties randomly until finishes transmitting of the full XML document.

A session will be established between the sender and the receiver until the end of transmission of the message, the RIDX communicator will pass the received message to the joiner module. The joiner module will stack the messages until it receives all the messages. A checksum will be made for each message received to check the integrity and safety of the messages. When receiving all messages, the joiner module will rejoin all the messages to build the XML file same as the original one, another validation check will be done to check again the integrity and safety of the combined XML file. The joiner module will inform the RIDX communicator to send a confirmation message to the source sender and then close the session.

Exceptions could happen during this whole process, these exceptions will be handled case by case in scenario that guarantees the safety and integrity of the XML data file.

• METHODS OF TRANSFER

At this stage, adapting one method for transferring data is preferable until this system reaches to a point of maturity, and then it can be integrated to other methods as needed. A major method which RIDX communicator will depend on is Point-to-Point communication protocol, a TCP/IP socket communication network is the backbone of the communication between any two nodes, the module which is responsible of handling this kind of communication is RIDX communicator module. Using java technology as a tool to create environment contains metamorphism and multi threaded socket handling, this will guarantee the communication in multi session environment at the same period of time in an efficient way.

The backbone of the communication carriers can be used to facilitate the communication between all RIDX parties, a leased line, an ATM connection, or thru internet connection are all possible carriers. In addition, a various security protocols such as Virtual Private Network is also adaptable, the decision of choosing the carrier or choosing the security protocol depends on the firm.

• TRANSPORT

The RIDX will include the following for transport methods:

1. RIDX communicator guarantee the delivery of the XML files
2. RIDX Shredder/Joiner guarantee the Integrity of a record level along with the file level, a checksum or MD5 techniques is used
3. Data encapsulation used to guarantee privacy of data
4. RIDX combines reliability and multicasting
5. Support for all file types binary, text etc. (Future Design)

5. RIDX PERFORMANCE TEST

A test conducted on Dell 1850 3 GHz with 4GB RAM on 2.6.9-14 RedHat Enterprise Linux 4 / 64 bit system, a Dell switch of 1GBps, SUN JVM 1.6.0_3. Each node (box) contains RIDX system, it will send number of M messages to all cloud nodes; receiver can be a sender at the same time, the timer will start when first message has been received, again the timer will stop when last message has been received, number (M) and size (S) of messages is known to all RIDX nodes. The computation of message rate in each RIDX with its throughput will be calculated by multiplying {number of senders N} times {number of messages M}. A configuration file contains same parameters for all RIDX nodes:

- N = 4 then 6 then 8, senders are same receivers, that means N equals to number of RIDX nodes also
- M = 1,000,000
- S = 1 then 2.5 then 5 KB the size of each message

6. EXPERIMENTAL RESULTS

The computation of message rate = (M * N) / T, where T is the total time of receiving all messages. Therefore, X & Y axis represents (S) and (Rate & throughput) simultaneously. For example: a collection of

data results when the test is done, we need to calculate the average rate of number of messages received per second, this can be done by summing up the data collected and divide it by (N), then we calculate the throughput by multiplying the (S * average message rate). As shown below in the graphs.
When compared TCP against UDP in RIDX, a TCP & UDP variance between four to ten nodes, we notice that the TCP begins at 49 MB per Second until it drops down to 34 MB per second, on the other hand, UDP begins at 56 MB per Second until it drops down to 51 MB per second, this shows that a significant better performance accomplished by UDP. But we also notice that when S = 2.5k, UDP drops down from 94 to 65 MB per second, and the same for larger message size; this is due to Maximum Transmission Unit system OS kernel parameter limitation to size of 1500, so whenever message size gets large and exceeds the Maximum Transmission Unit size, then the UDP packet split into more IP packets, , reasoning more delays to following packets, this problem we do not find it in TCP protocol because it creates IP packets which is always under than Maximum Transmission Unit size.

7. CONCLUSIONS

As shown in the results, RIDX performance is good in a group of four to ten nodes. Despite the UDP performance over TCP, but as mentioned above, the purpose of RIDX architecture is to provide security, reliability and scalability. Moreover, this architecture can be tailored by any environment using its variety of useful customized stack of protocols.

REFERENCES

[1] Curbera, F. Duftler, M. Khalaf, R. Nagy, W. Mukhi, N and Weerawarana, S.: Unraveling the web services web: An introduction to SOAP, WSDL, UDDI. IEEE Internet Computing, 6(2): 86-93, March-April 2002.
[2] Rabhi, F.A. and Benatallah, B.: An integrated service architecture for managing capital market systems. IEEE Network, 16(1):15-19, 2002.
[3] Kohloff, Christopher and Steele, Robert: Evaluating SOAP for High Performance Business Applications: Real-Time Trading Systems, 2003,
ystems,2003,http://www2003.org/cdrom/papers/alternate/P872/p872\kohl
hoff.html, accessed 22 March 2005.
[4] FIX Protocol Ltd. FIXML: A markup language for the FIX application message layer. http://www.fixprotocol.org/WORKGROUPS/928951581/wpaper.html, accessed 8 June 2002.
[5] FIX Protocol Ltd. The Financial Information Exchange Protocol (FIX), version 4.3, August 2001. http://www.fixprotocol.org/specification/fix-43-pdf.zip, accessed 8 June 2002.
[6] F. E. Bustamante, G. Eisenhauer, K. Schwan, and P. Widener. Efficient wire formats for high performance computing. In Proceedings of the 2000 Conference on Supercomputing, 2000.
[7] D. Davis and M. Parashar. Latency performance of SOAP implementations. In Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, pages 407-412, 2002.
[8] M. Govindaraju, A. Slominski, V. Choppella, R. Bramley, and D. Gannon. Requirements for and evaluation of RMI protocols for scientific computing. In Proceedings of the 2000 Conference on Supercomputing, 2000.
[9] FIXML – FpML Overview of the schema FIXML4.4, http://www.fixprotocol.org/specifications/fix4.4fixml-10 October 2006,
[10] zur Muehlen, M.; Nickerson, J.V.; Swenson, K.D.: Developing Web Services Choreography Standards – The Case of REST vs. SOAP. Decision Support Systems 37 (July 2005), Elsevier, North Holland.
[11] Pautasso, Cesare; Zimmermann, Olaf; Leymann, Frank (2008-04), "RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision" (HTML), 17th International World Wide Web Conference (WWW2008) (Beijing, China)
[12] R. Fielding. A little REST and Relaxation. The International Conference on Java Technology (JAZOON07), Zurich, Switzerland, June 2007.
[13] J. Arturo Pérez, et. Al. "Quality of Service Analysis of IPSec VPNs for Voice and Video Traffic," Advanced International Conference on Telecommunications, Guadeloupe, (February 2006).
[14] R. Wagner, et. Al. “Recommendations for Infrastructure Protection, 2006,” Gartner Group, (February 2006).
[15] J. Yu and C. Liu, "Performance Analysis of Mobile VPN Architecture," 4th Annual Conference on Telecommunications, and Information Technology, Las Vegas, (March 2006) [2] “At last Internet Banking takes off.”, Osterland, Andrew, ABA Banking Journal, Jul99 Issue 7, p63, 1p.

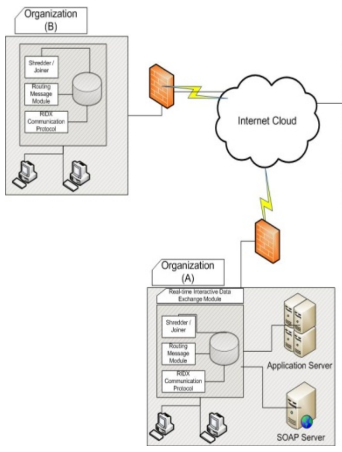
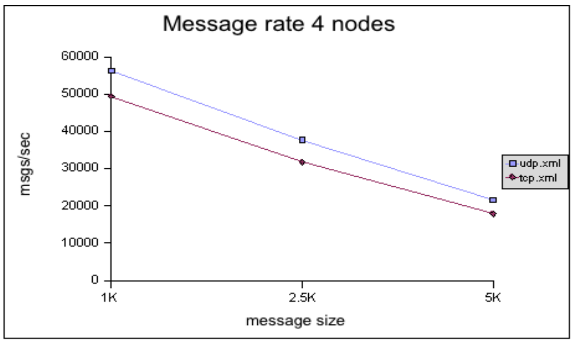
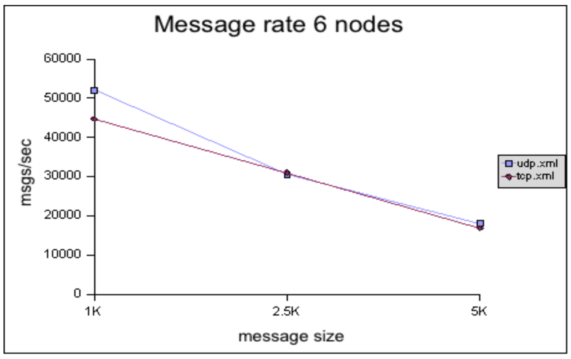
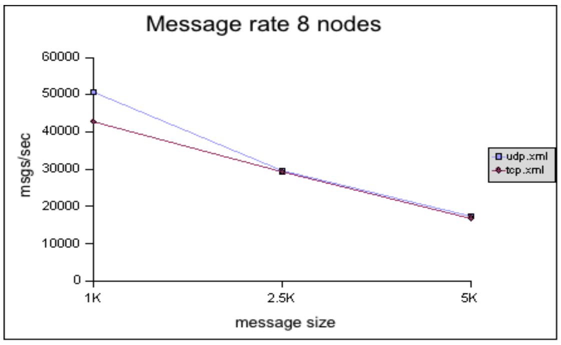
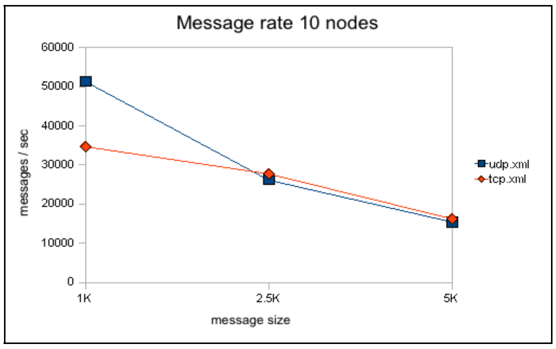


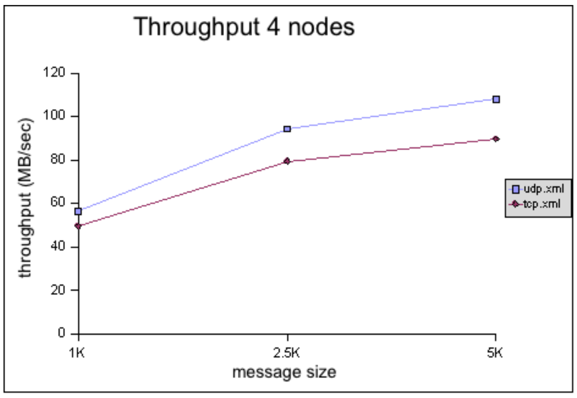
Figure 1.0 illustrates the network general structure between Different RIDX

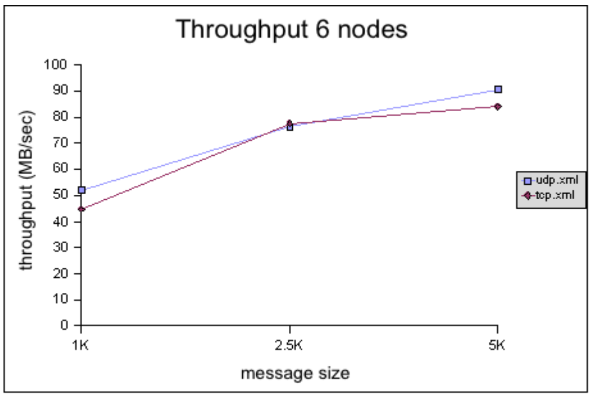


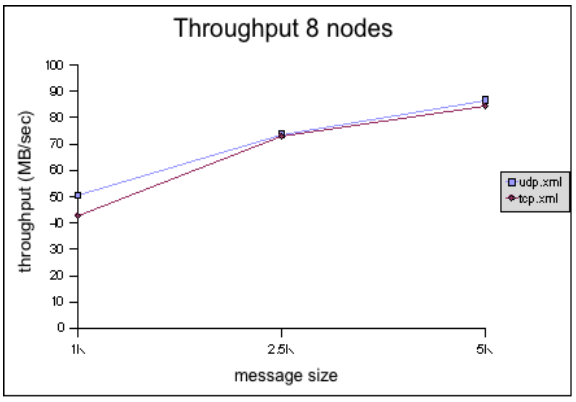


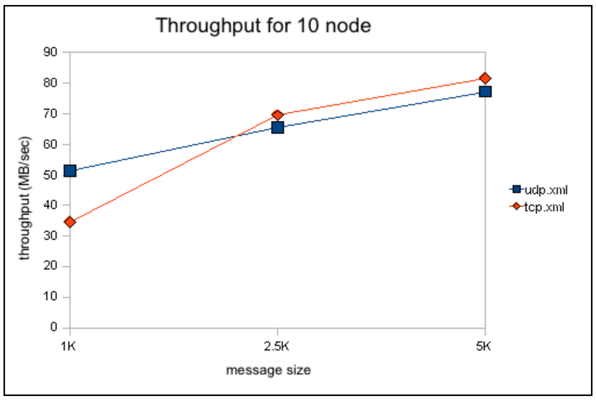


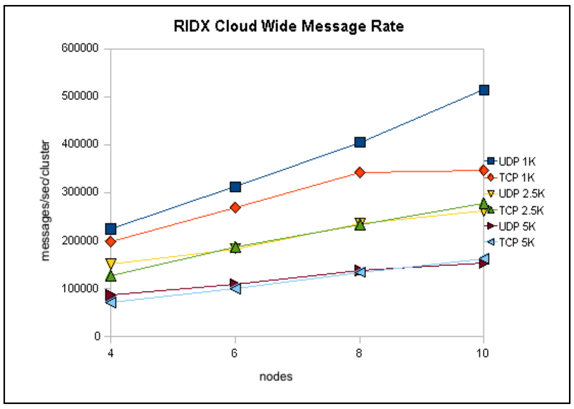


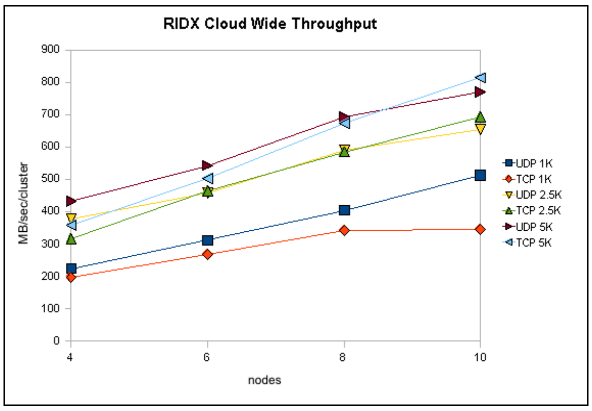





Throughput 8 nodes
throughput (MB/sec)
message size
1K
2.5K
5K
udp.xml
tcp.xml


Throughput for 10 node
throughput (MB/sec)
message size
1K
2.5K
5K
udp.xml
tcp.xml


RIDX Cloud Wide Message Rate
messages/sec/cluster
nodes
UDP 1K
TCP 1K
UDP 2.5K
TCP 2.5K
UDP 5K
TCP 5K


RIDX Cloud Wide Throughput
MB/sec/cluster
nodes
UDP 1K
TCP 1K
UDP 2.5K
TCP 2.5K
UDP 5K
TCP 5K