ISSN: 2320-1363

Management of Trust worthy coordination of Web Services Business Activities Using Lightweight BFT

G.Mamatha *1, T.Subhashini *2, R.Pitchaiah *3

1*M.Tech Scholar, Dept of CSE, Universal College of Engineering & Technology, Perecherla, Dist: Guntur, AP, India

2*Assistant Professor, Dept of CSE, Universal College of Engineering & Technology, Perecherla, Dist: Guntur, AP

3*Associate Professor & HOD, Dept of CSE, Universal College of Engineering & Technology, Perecherla, Dist: Guntur, AP, India

Abstract:

Many organizations use multiple software systems for management. Different software systems often need to exchange data with each other, and a Web service is a method of communication that allows two software systems to exchange this data over the internet. Providing security to the data and handling unwanted threats to the web services are major challenges in Web service business activities. We are proposing a light weight Byzantine Fault tolerant algorithm (BFT) for providing the trusted coordination between multiple participants and initiator in a web service Business Activities. The BFT algorithm considers all possible sources like source ordering, rather than total ordering, of incoming requests to achieve Byzantine fault tolerant, state-machine replication of the WS-BA coordination services.

Keywords: Web Services, Business Activity, Distributed Transaction, Byzantine Fault Tolerance, Service Oriented Architecture.

1. Introduction:

Service-oriented computing and the Web services technology changes the nature of internet usage and widely improves the importance and usage of distributed computing which leads to rapid growth of Business activities conducted online. The Web services business activity (WS-BA) specification [2] has recently been ratified by OASIS to standardize the activation, registration, propagation and termination of Web services business activities. However, WS-BA does not specify a standard way for an initiator to communicate with the WSBA coordinator. Even though this design choice makes it easy for vendors to integrate WS-BA coordination functionalities into their business process engines, it is difficult to ensure interoperability among initiators and coordinators from different vendors [3].

ISSN: 2320-1363

Since a business activity almost inevitably will involve multiple enterprises, expecting or requiring all such enterprises to use and trust a single vendor's product seems to be unrealistic. To address this issue, Erven et al. [6] proposed an extension to the WS-BA specification, referred to as Web Services-Business Activity-Initiator protocol (WS-BA-I) to enable the separation of the coordination functionality and the business logic, and to standardize the interactions between the initiator and the coordinator. This extension makes it possible for a third party to offer coordination services to enterprises who wish to conduct Web services business activities with minimum modification to their workflow engines. Obviously, in order for such coordination services to be widely adopted, they must ensure high degree of security and dependability. This work is aimed to provide a lightweight solution to enhance the security and dependability of the WS-BA coordination services.

2. Web Services Business Activity Architecture:

Web Services Business Activity posses the basic elements called initiator, Coordinator and a set of participants. All business activities are started and terminated by the initiator. The initiator also propagates the business activity to other participants (through a coordination context included in the requests to other participants). The outcome of the business activity is determined by the initiator according to its business logic. The Figure 1 shows all the components of WS-BA.

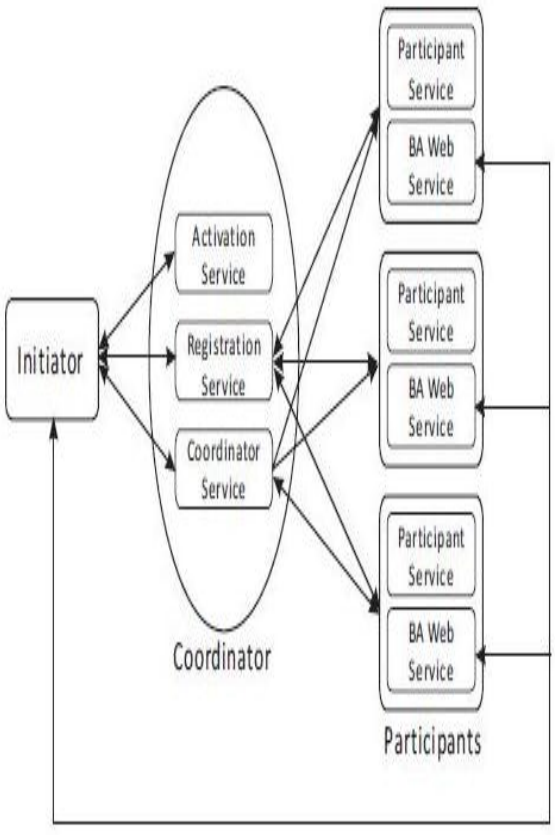

Figure 2.1: Web Services Business Activity

WS-BA specifies two coordination types, Atomic-Outcome and Mixed-Outcome, and two coordination protocols used between the coordinator and a participant, Business-Agreement with Participant-Completion (BAwPC) and Business-Agreement-with-Coordinator-Completion (BAwCC). Each participant registers either one of the two protocols, which are managed by the coordinator of the business activity. During the entire process of Web service Business activity the communication takes place by using a set of messages like Completed, Close, Compensate, Failed, Cannot

ISSN: 2320-1363

Completed, not Completed, cancelled.WS-BA provides two types of services called coordinator-side services and a participant service at the participant-side. The coordinator-side services include Activation, Registration and Coordinator services, and they run in the same address space. For each business activity, all but the Activation Service are provided by a (distinct) coordinator object.

3. Threat Analysis:

There are different ways to affect the WS-BA system by the threats. The Threats which effect and change the behavior of the system can be come from any of the sources like initiator, coordinator and participants.

3.1. Threats from a Faulty Coordinator

The Threats are majorly come into picture in Faulty Coordinator are because of un trusted activation services, Wrong registration services and Wrong Coordination services.

3.2. Threats from a Faulty Participant

The Threats are majorly come into picture in Faulty Participant are because of the coordinator and Initiator. A faulty participant could (1) lie about its execution status, and send the coordinator reports inconsistent with its internal state, and (2) send conflicting reports to different coordinator replicas. The initiator insists on the use of digital signatures with all messages exchanged with the participants, and logs them. It is conceivable for the initiator to give preference to the participants who offer higher degree of quality of services that minimize such threats.

3.3. Threats from the Initiator

Since a business activity is always initiated, and its progress and outcome is controlled, by the initiator, the integrity of the business activities that originated at a Byzantine faulty initiator cannot be guaranteed. Similar to the circumstances for a faulty participant, we can prevent this from happening by replicating the initiator and employing BFT techniques.

4. The Light Weight BFT:

The Light weight Byzantine Fault tolerant algorithm (BFT) algorithm is mainly used to address and eliminates the threats came from various sources like coordinator, Participant and Initiator. Lightweight BFT algorithm fulfills the following two objectives:

O1. If a request is delivered at a correct replica, it must eventually be delivered at all correct replicas according to the sender's source order.

O2. In the event of a faulty client sending conflicting requests (that carry the same sequence number) to correct replicas, only one of the requests may be delivered to all correct replicas, if at all.

We assume that $3f+1$ coordinator replicas are available, among which at most f can be faulty. The initiator and the participants are not replicated (our algorithmcan be trivially modified if these entities are in fact replicated). There is no limit on the number of faulty participants. The initiator could be faulty as well. Each

ISSN: 2320-1363

coordinator replica is assigned a unique id *k*, where *k* varies from 0 to 3*f*. We assume that the algorithm operates in an asynchronous distributed environment. However, we assume that the network is reliable. In particular, if a correct participant/initiator sends a message to a correct coordinator replica, the message will reliably arrive at the replica eventually. The same is true for messages exchanged between correct replicas. This assumption can be easily satisfied by using TCP communication, or by using the mechanisms defined in the Web Services Reliable messaging (WS-RM) standard.

All WS-BA messages, *i.e.*, the messages exchanged between the coordinator and the initiator and those between the coordinator and the participants, carries a monotonically increasing sequence number. The sequence number is unique only within the respective connection between the coordinator and its client (*i.e.*, the initiator or the participant), but each connection is uniquely identified. For each connection, the first request is assigned a sequence number 0, and the sequence number is incremented for each subsequent request. The reply, if any, carries a sequence number matching that of the corresponding request. The same holds true for requests sent from the coordinator to the participants in the BAwCC or BAwPC protocols. Note that all messages defined in BAwCC and BAwPC protocols are one way messages.

All messages between the coordinator and participants are time stamped (to prevent the replay attack) and digitally signed (to ensure accountability and to prevent spoofing).We assumes that the coordinator replicas, the initiator, and the participants each has a public/private key pair. The public key is known to all of them, while the private key is kept secret to its owner. We assume that the adversaries have limited computing power so that they cannot break the digital signatures created by non-faulty entities.

The normal operation of the BFT algorithm is shown in Figure 2 (for $f = 1$). A client (*i.e.*, the initiator or a participant) sends the request to all coordinator replicas. The request has the form <REQUEST, *s*, *o*> σc, where *s* is the sequence number, *o* is operation to be invoked (including the coordination context if needed) and σc is the signature of the message signed by the client.

Upon receiving a request, a replica verifies the signature, and checks the validity of the requested operation and if the sequence number carried with the request matches the next expected sequence number. A replica broadcasts a commit message to all other replicas if the request passes the verification. The commit message has the form <COMMIT, *k, s, d*>σk, where *d* is the digest of the request message, and *k* is the replica number. Note that all replicas play an equal role, in particular, no replica acts the primary since total ordering is not needed (*i.e.*, the sequence number is assigned by the client instead of the primary).

When a replica receives both the request and $2f$ matching commit messages from other replicas, it delivers the request and makes the state transition. The reply, if any (recall that many WS-BA messages are one way), has the form <REPLY, *k, s, r*>σk, where *r* is the response. Some operation

ISSN: 2320-1363

might trigger the sending of nested requests to the participants (*e.g.*, the "Complete" instruction from the initiator will cause the coordinator to send the same command to the designated participant). Such nested requests (not shown in Figure 2) have the form <NESTED REQUEST, *k*, *s*, *o*>*σk*, *which* is similar to that of regular requests with an additional field indicating the sending replica number.

If a replica receives a commit message before it receives the referenced request, it requests a retransmission of the request from the replica that sent it the commit message.

If a replica receives $f + 1$ consistent commit messages, but the digest in the commit message is different from the request it has received, it requests a retransmission of the missing request from the $f + 1$ replicas, logs the event, and abandon the original request.

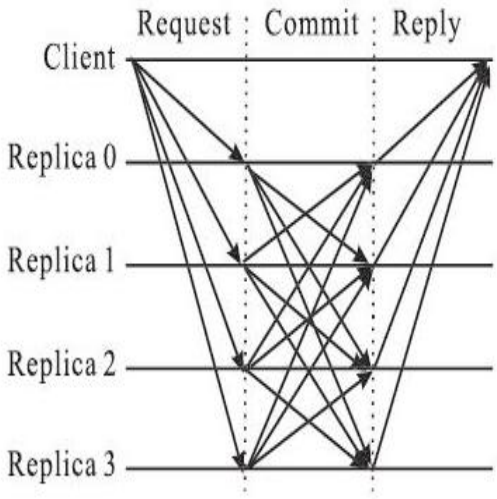

Figure 4.1 Normal Operations Of Light weight BFT with *f*=1

If the client (*i.e.*, the initiator or the participant) that issued the request expects a reply, it must collect at least $f + 1$ matching replies sent by different replicas before it delivers the reply. The same mechanism is used for the participants to handle (nested) requests issued by the coordinator replicas. By collecting $f + 1$ matching replies, it is guaranteed that at least one of them is sent by a correct replica.

5. Conclusion and Future Work:

We proposed a new algorithm called light weight Byzantine Fault tolerant algorithm (BFT) for providing the trusted coordination between multiple participants and initiator in a web service Business Activities. The BFT algorithms efficiently detect the dangerous threats from various sources like initiator, coordinator and participants. The BFT algorithm enhances the Business activities by eliminating all the threats. The lightweight BFT algorithm that avoids most of the runtime overhead associated with generic BFT algorithms.

References:

[1] Hua Chai, Honglei Zhang, Wenbing Zhao, P. Michael Melliar-Smith, Louise E. Moser "Toward Trustworthy Coordination of Web Services Business Activities" IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 6, NO. 2, APRIL-JUNE 2013

[2] T. Freund andM. Little. *Web Services Business Activity Version 1.1, OASIS standard*, April 2007.

[3] H. Erven, H. Hicker, C. Huemer, and M. Zapletal. The Web Services-

ISSN: 2320-1363

BusinessActivity-Initiator (WS-BA-I) Protocol: an extension to the Web Services-BusinessActivity specification. In *Proceedings of the IEEE International Conference on Web Services*, Salt Lake City, Utah, July 2007.

[4] D. Davis, A. Karmarkar, G. Pilz, S. Winkler, and U. Yalcinalp, Web Services Reliable Messaging (WS-ReliableMessaging) Version 1.1, OASIS Standard, Jan. 2008.

[5] D. Dolev, "The Byzantine Generals Strike Again," J. Algorithms,vol. 3, no. 1, pp. 14-30, 1982.

[6] D. Dolev and H.R. Strong, "Authenticated Algorithms for Byzantine Agreement," SIAM J. Computing, vol. 12, pp. 656-666, 1983.

[7] H. Erven, H. Hicker, C. Huemer, and M. Zapletal, "The Web Services-BusinessActivity-Initiator (WS-BA-I) Protocol: An Extension to the Web Services-BusinessActivity Specification," Proc. IEEE Int'l Conf. Web Services, pp. 216-224, July 2007.

[8] M. Feingold and R. Jeyaraman, Web Services Coordination, Version 1.1, OASIS Standard, July 2007.

[9] T. Freund and M. Little, Web Services Business Activity, Version 1.1,OASIS Standard, Apr. 2007.

[10] H. Garcia-Molina, F. Pittelli, and S. Davidson, "Applications of Byzantine Agreement in Database Systems," ACM Trans. Database Systems, vol. 11, no. 1, pp. 27-47, 1986.

[11]Y. Gil and D. Artz, "A Survey of Trust in Computer Science and the Semantic Web," J. Web Semantics, vol. 5, pp. 58-71, 2007.

[12] T. Grandison and M. Sloman, "A Survey of Trust in InternetApplications," IEEE Comm. Survey Tutorials, vol. 4, no. 4, pp. 2-16,Oct.-Dec. 2000.

[13] .K. Iwasa, J. Durand, T. Rutt, M. Peel, S. Kunisetty, and D. Bunting, WS-Reliability 1.1, OASIS Standard, Nov. 2004.

[14] K.P. Kihlstrom, L.E. Moser, and P.M. Melliar-Smith, "The SecureRing Group Communication System," ACM Trans. Information and System Security, vol. 4, no. 4, pp. 371-406, Nov. 2001.