Volume-4, Issue-6, June-2015 • ISSN No 2277 - 8160

Research Paper Commerce

Responsible organization of Pay Services

SAPNA PANCHAL

ABSTRACT

A milestone in e-payment through online is secured cloud environment. The trust of transaction lies in context in which it is started and completed. We have proposed a fault tolerant environment for e-payment at merchant node. A light weight Byzantine algorithm is implemented to make the process of transaction reliable. The user can search the bank details in a distributed environment in source ordering mode. A single user can access their bank accounts in different providers using a universal pay identity number. The proposed prototype is a composition of web services provided by different bank services. For secured communication the message are encrypted with the sophisticated RSA algorithm. For reliability we deployed Byzantine fault tolerance environment. The application may provide the commercial users an easy and secured mode of e-payment. It also reduces the overhead for infrastructure maintenance when deployed as Infrastructure as Service (IaaS) in cloud.

KEYWORDS : Lightweight Byzantine Algorithm, Web Services, Scalability, fault-tolerance, SOA, Payment Service, secured transaction.

I. INTRODUCTION

In world of virtuality, all the trading and commercial transactions should happen in a protected environment since the trading are realised in an electronic mode. The service providers take more interest to make the e-transaction in a secured way. As defined by Wikipedia, A Web service is a method of communication between two electronic devices over World Wide Web. The access through the services is provided through an Application Programming Interface (APIs).

The latest trend is to combine different web services offered by third parties for business activities and provide them as web service composition to reduce the overhead of maintaining a heavy infrastructure (IaaS). The main point to ponder at this stage is "trust" [12] of the services. The user should be provided with a fault tolerant environment which will not compromise due to any kind of failures like machine failure, malicious behaviour of the application or delink due to communication fault. A trust of e-payment lies in the completion of the task in the context in which it has been initiated. For example, if a user selects a particular airline for booking, the service should not provide another airline ticket. This is a distrust which is done to the user. This should not happen in money transaction, (i.e) a service should not deposit the cash in the account which is not selected by the user.

We require a fault tolerant environment to deploy the service which was proposed by W. Zhao [2] in the Toward Trustworthy Coordination of Web Services Business Activities. The proposed system has a coordination of services provided by banks. A single person holds various types of account in various banks. The main transaction between the customer and the bank is credit and debit transaction. Web service coordination can be provided by creating an interface for the services to enhance the transaction. The coordination service provides the fault tolerant light weight environment by source ordering of the request [2]. This is used for searching the bank accounts and for listing the account details like name, account number, bank name and balance which satisfy the given condition.

The study addresses the following situations. The first issue is a fault tolerant environment, which is always available with an expected behaviour. A byzantine fault tolerant environment [4] is developed using 3f+1 replicas of the coordinator service, among which f can be faulty at any instance of time. The fault can be due to arbitrary behaviour of the server replicas due to malicious code, server failure or compromise of the replica as addressed in W. Zhao [2]. For f=1, 4 replica and for fault of 2, 7 server replica should be provided. The second issue is trust of completion of the service, which is achieved by deploying light weight byzantine algorithm in 3f+1 replica. The algorithm is handled by the coordinator replica in an asynchronous distributed environment [2]. The replica can be placed in same address location or at different geographical positioning. The algorithm uses source ordering of the request instead of total ordering. In source ordering the request from the same source are numbered and replies acknowledge the number of the request.

The suggested environment suits well for e-payment for a customer at merchant node using a single id. It is a combined activity of distributed search and atomic transaction [2] and [3]. The user should register all their account detail of different bank with the universal pay service which provides a universal pay identity number (UPI). The same can provided as a single card which hold the UPI, customer detail and encrypted key. For authentication, OTP is provided during the search and payment. The completion of the payment service is approved by the transaction id sent to user.

II. RELATED WORKS

A Muti-Master type server replica which has been handled in Byzantine Fault Tolerant Coordination for Web Services Atomic Transactions [2], they have used atomic transaction mode creating a transaction coordinate context. In Byzantine fault tolerance [3], for a fault of 1, it needs at least 4 replicas and for fault of 2, it need at least 7 replicas (i.e) 3f+1. The coordinator collects at least 2f or f+1 replica replies with 2 sets of quorum [11]. The state of the request is marked as given in [9]. The Byzantine always causes arbitrary failure behaviour of server replicas. The fault can be caused due malicious code, machine fault or the replica would compromise its behaviour which is proved in [10]. The light weight byzantine algorithm handles the request in source ordering which does not require inter communication messages. In the trustworthy of net banking [13], the accounts of different banks are connected to perform online transaction by the user in atomic mode. In the study, source ordering of request [2] in a distributed environment is used which is stateless.

The prototype includes two types of operations: search and payment. The initiation can be a client machine or an instrument used by the merchant at the purchase point. The suggested prototype satisfies the issues addressed by M.Swientek et. al in [14]. At every stage of process we have used OTP to ensure the authentication of the user. The codes are generated with RSA algorithm to maintain reliability. The request-response is done with SOAP protocol description.

III. COORDINATION OF BANK SERVICES

The third party (i.e) universal pay service coordinates different bank payment services. The account holder registers with the universal pay service and gets a card or ID. An OTP is provided for both search and when the transaction is initiated. The transaction reflects at both the end. The coordinator service is deployed in four different machines (3f+1) and the LBFT algorithm implemented that makes the environment fault tolerant and trustworthy.

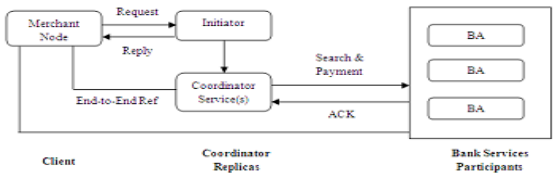


Fig. 1 Process diagram for the proposed pay service

The user can perform two types of operation search and pay service. When the initiator receives a search, it expects the universal pay id, amount and OTP to be searched. The detailed work is described below.

1. Creation of Universal Pay ID.

As a prerequisite the customer should register their bank account details with the universal pay service. A unique id is generated and for real time purpose the same can be provided as card. The card can have the encrypted details about the customer.

Fig. 2 Registration Process of Universal Pay Service

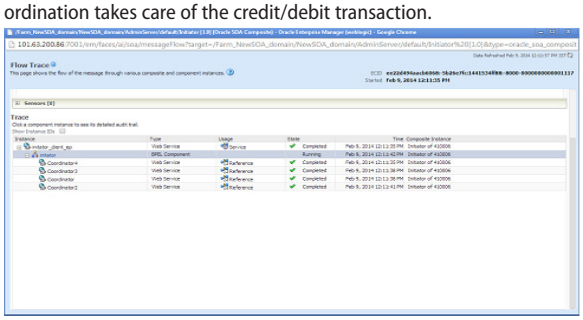
An OTP is sent to user to verify the process initiation. When the user confirms, the coordinator-registration is completed. After the search is completed, the client can view the list of bank which matches the balance (i.e) the end point reference.

2. Search and Pay

When the search is initiated, an OTP is sent the customer. For real-time purpose the user can generate their OTP before transaction using mobile apps or a device which generates OTP. The search will proceed only when the OTP is entered. We used SOAP protocol to send and receive message and deployed the structure in a commercial SOA architecture. After search output, the user selects the bank, as per the bank procedure, the user may enter a PIN or follow the mechanism provided by the concern bank to proceed the transaction process. It satisfies the trust and also gets the acknowledgement from the user for the transaction.

IV. RESULT AND DISCUSSION

The experiment is tested on a prototype deployed in a SOA and using the necessary encryption and authentication procedure. The user accounts are registered with a universal pay number. During search it searches the account with this id, matching balance and sends the reply. The balance will be always greater than or equal to the given requirement. The pay service which is new service provided in the coordination takes care of the credit/debit transaction.



The proposed system can be deployed where the customer uses debit or credit accounts. Transaction is conducted with the context in which the client has initiated the action.

1. Creation of OTP

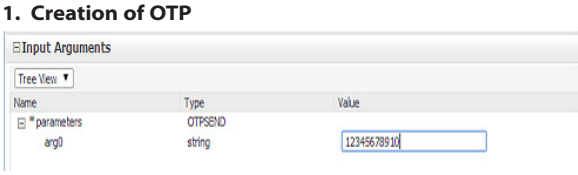


Fig 4 : Creation of OTP.

The user gives the UPI for generating the OTP. The OTP is sent to the mail id or the registered mobile number. The OTP is generated for every individual transaction.

2. Search Bank Details

Fig 5 : Search operations with OTP.

As soon as the OTP is received the user enters it along with the amount. The search bank service retrieves the bank(s) which has balance more than or equal to the given amount and shows only those bank(s) to the user.

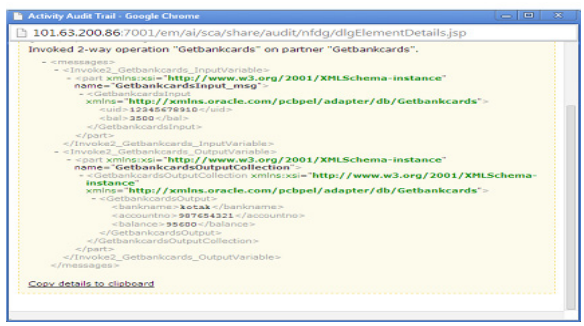


Fig 6 : Output of search operation.

3. Payment

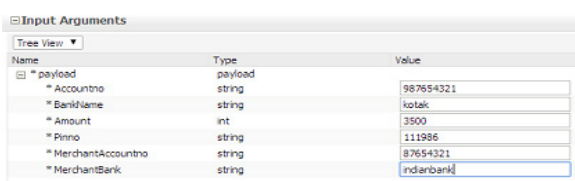


Fig 7 : Payment process initiation.

The user selects one bank for the payment processing. As authentication mechanism we have used PIN number of that bank to complete the transaction.

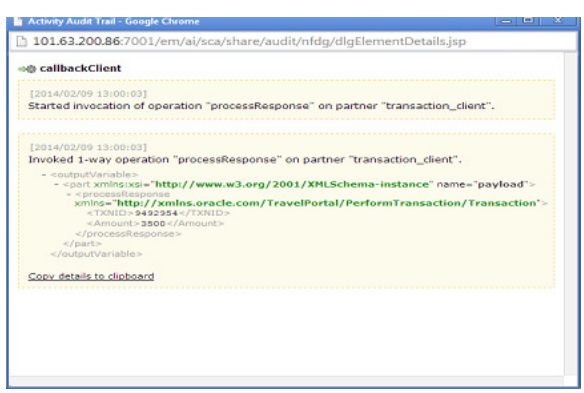


Fig 8 : Completion of Payment process

As the completion of transaction a unique transaction id is generated. Thus a secured transaction has been completed in a fault tolerant environment with proper authentication. The system is a totally secured one even if the customer looses the UID card; other person cannot perform any transaction. The machine critical situation is also handled by replicating the coordinator service and made fault tolerant. The transaction is completed with the context in which it has been initiated.

V. CONCLUSION

The environment which we used is fault tolerant and also performs the transactions trustworthy by successfully deploying the light weight Byzantine algorithm for universal pay service at merchant node. We have not considered any concepts on the database part. We have successfully implemented a trustworthy algorithm in a fault tolerant environment and combined distributed and atomic activities together.

REFERENCES

[1] http://docs.oracle.com/cd/E16764_01/integration.1111/e10224/toc.htm | [2] W. Zhao and H. Zhang, "Toward Trustworthy Coordination of Web Services Business Activities", IEEE Transactions on Services Computing, Vol. 6, no. 2, April-June 2013. | [3] W. Zhao (2007A), "Byzantine Fault Tolerant Coordination for Web Services Atomic Transactions", Proceedings of Fifth International Conference on Service-Oriented Computing, Pp. 307-318. | [4] Giuliana Santos Veronese, Miguel Correia, Alysson Neves Bessani, Lau Cheuk Lung, and Paulo Verissimo, "Efficient Byzantine Fault-Tolerance", IEEE Transactions on Computers, vol. 62, no. 1, January 2013. | [5] WU Nai-zhong, "Dynamic Composition of Web Service Based on Cloud Computing", International Journal of Hybrid Information Technology Vol.6, No.6, pp.389 - 398, 2013. | [6] Torsten Hahmann , "Workflow/Web Service Composition", Vorbereitungs seminar Bachelorprojekt ASG SS 2005. | [7] Schahram Dustdar and Wolfgang Schreiner, "A Survey on Web Services Composition", Int. J. Web and Grid Services, Vol. 1, No. 1, 2005. | [8] Hannes Erven, Georg Hicker, Christian Huemer and Marco Zaptletal, "The Web Services-BusinessActivity–Initiator (WS-BA–I) Protocol: an Extension to the Web Services-BusinessActivity Specification", IEEE International Conference on Web Services (ICWS 2007). | [9] W. Zhao (2007), "A Byzantine Fault Tolerant Distributed Commit Protocol", Proceedings of 3rd IEEE International Symposium on Dependable, Autonomic and Secure Computing, Pp. 37-44. | [10] T.Grndison and M. Sloman, "A Survey of Trust in Internet Applications", IEEE Comm., Survey Tutorials Vol 4, No 4, PP 2- 16 Oct – Dec 2000. | [11] Miguel Castro and Barbara Liskov, "Practical Byzantine Fault Tolerance and Proactive Recovery", ACM Transactions on Computer Systems (TOC), Volume 20 Issue 4, November 2002, Pages 398 – 461. | [12] T. Grandison and M. Sloman, "A Survey of Trust in Internet Applications," IEEE Comm. Survey Tutorials, vol. 4, no. 4, pp. 2-16, Oct.-Dec. 2000. | [13] Supriya Kurian, Ramya. G. Franklin, " Trustworthy Coordination of Web Services Atomic Transaction for Net Banking" The SIJ Transactions on Computer Science Engineering & its Applications (CSEA), Vol. 1, No. 1, March-April 2013 | [14] M.Swientek, U,Bleimann and P.S Dowland, " Service-Oriented Architecture : Performance Issues and Approaches, Proceedings of the Seventh International Network Conference (INC 2008). |