Journal of Computer Science 10 (3): 521-529, 2014
ISSN: 1549-3636

doi:10.3844/jcssp.2014.521.529 Published Online 10 (3) 2014 (http://www.thescipub.com/jcs.toc)

PERMUTATION OF HTTPI AND HTTPS IN WEB SERVICES AGAINST ATTACKS FOR SECURITY ENHANCEMENT

Chakaravarthi, S. and V. Ramachandran

Department of Information Science and Technology,
Faculty of Information and Communication Engineering, Anna University, Chennai, India

Received 2013-08-20; Revised 2013-11-26; Accepted 2013-11-30

ABSTRACT

The Hyper Text Transfer Protocol (HTTP) protocol plays a vital role in Web Services Security. Though the HTTPs provide excellent security, they are not flexible enough to allow caches. HTTPi provides high integrity and low security whereas HTTPs provide low integrity and high security. The goal of WS activity is to build up set of technologies in order to direct WS to their complete prospective application. WS play an excellent role, without which the internet applications cannot be made. To provide both high security and high integrity in Web Services (WS), a new model is proposed. In this model, the combined HTTPs's security and HTTPi's flexibility are considered to provide the best WS. In addition, the user affords the self encrypted data for privacy preserving the requester agent. Finally, the requester agent encrypts the particular data so as to create two protections known as self protection and agent protection. Due to mounting threats in the WS, numerous developers and researchers attempt to enhance additional safekeeping in service level. When WS usages are constantly increasing, it is necessary to give proper security as well as flexibility in WS. The requester agent sends the data to the next level. In this message level, the header information can be self verified through appropriate security mechanism proposed in the model. The results of this proposal are compared with the existing methods and better performance is obtained by calculating the throughput and response time.

Keywords: HTTPi, HTTPs, Privacy Preserving, Web Services

1. INTRODUCTION

At present, the WS provide the feasibilities to interact one machine to another machine. It is playing enormously superior position in World Wide Web. There are five technologies in WS to be exact as HTTP, Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), Universal Description Discovey and Integration (UDDI). In the actual working on the WS, these technologies coupled mutually and they provide integration to interconnect with one machine to another machine. WS works taking part of the transitional websites as the model works on the perception of entity-entity connectivity. Due to mounting threats in the WS numerous developers and researchers enhancement to give additional safekeeping in service level.

XML is one the markup language for documents surrounded structured information. It is one of the foundations designed for the WS. XML has prepared three services that are described as describing, discovering and invoking. The WS security supported on XML and XML schema. When a huge sizes of documents the 'XML text based document' are supporting.

For securing WS, it has to consider five essential areas; that is communication level security, communication privacy, parameter inspection, authentication and authorization. The XML conceptual in WS like a stack service. The **Fig. 1** shows that the first level service networking is providing communication between sender and receiver, this communication will take place through HTTP, HTTPs, FTP, IIOP. The second level is the XML-Based messaging through SOAP protocol.

Corresponding Author: Chakaravarthi, S., Department of Information Science and Technology, Faculty of Information and Communication Engineering, Anna University, Chennai, India

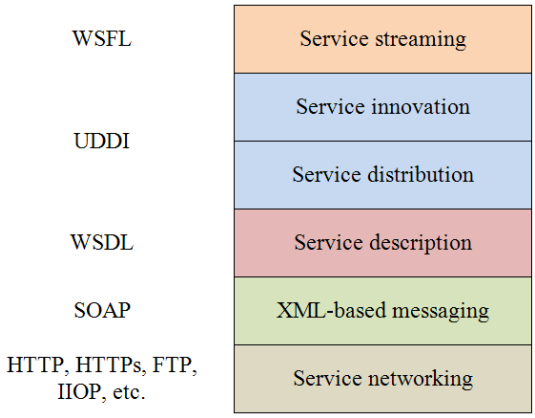

Fig. 1. Stack of Web Services

It is set of procedures it will carry and exchanging the communication each others. The third stage of the service stack is description of the service through WSDL; this stage provides signatures of the methods. The next stage the UDDI service is provides service distribution and service innovation. The last level through WS Flow Language (WSFL) is providing service to receiver.

Open application is the emerging technique. But it is not a confidential. From side to side authentication and data integrity is growing rapidly. The latest protocol HTTPi is ensures the entire security requirements through open applications. It is well-suited for cache proxies. The HTTPi is providing directed client-server authentication and integrity, but HTTPi is not concentrating confidentiality. An authentication is provided that username/password and binary tokens. The integrity is providing XML digital signature i.e., RSA-SHA1. The HTTPi is giving privileged throughput. Hence the main theme is touching to offers three types of securities like WS authentication, WS integrity and WS confidentiality (Choudhary and Nirmal, 2013).

The HTTPs used to prevent Man-In-the-Middle (MIM) attacks and Impersonate Web (IW). The HTTPi protocol is providing professional design and easy to arrange in web. Through Internet Explorer in client part using 'IE's Asynchronous Pluggable Protocol Extension Mechanism' the author constructed an end-to-end model to assess HTTPi and the server part in IIS 7 and server part modifications, but the author did not changed intermediate nodes. Hao *et al.* (2011) has proposed remote data integrity checking protocol in a cloud computing isolated data reliability and also he has provided feasibility for avoiding third party verification. For instead of giving third party verification i.e., un-trusted server for own self verification is providing reliability (Hao *et al.*, 2011; Xu *et al.*, 2013).

Jian (2011) is designed a WS security based on water-making techniques. Through SOAP service, the service suppliers and recipients are exchanging the information, while exchanging message digital signature and XML encryption technology is making guarantees. The way the author has taken to provide security in WS water mark technology (Zhang, 2011). The author with the help of WSDL, SOAP and UDDI provided flexible solution for problem of application integration. He applied the security three service levels likewise service security, service composition and service semantics, in a semantics WS SOAP is playing good role, hence the plan is to use SOAP in the proposed work to provide better performance. The author Nicholas expressed his view agent based WS system concert assorted streams in order to afford situational attentiveness potential. In a semantic web grid message transport layer SOAP message is providing security in high level, for the work in a SOAP message level requester agent encrypted data transferring (Hao *et al.*, 2011).

In a WS system Quality of Service (QoS) represents like delivery, deadline, quality of products, cost of service, through put of service completion as well as extended their services. Once the above aspect is obtainable to reduce in a business organization the QoS metrics directly disturbing business. In his study mainly concentrated time, cost and reliability, for implementing QoS he or author developed SWR algorithm (Rathore and Suman, 2011).

The **Fig. 2** shows the request agent to provider agent process which is described latter. The process and its relevant steps are explained in **Fig. 3**. The main purpose of proposed work is to provide enhanced security in the requester side and responding side. In the experimental program has been calculated average response time, average throughput and reply size per request with different types of scenarios like Non Secure, less Confidentiality (A), Highly Secure, non supportable for cache proxies, confidentiality (B), Secure, non Confidentiality, supportable for cache proxies(C), Combination of both highly securable and supportable for cache proxies (D) is tabulated in **Table 1** and corresponding graph is shown in **Fig. 4**.

The **Table 2** shows that web services scenarios Vs average response time, average throughput and reply size per request and protocols HTTP, HTTPs Vs Response time, reply Size. It's clearly explains combinations of HTTPs and HTTPi provides more secure and most cache accessibility in the web services. The results obtained are plotted in **Fig. 5**.

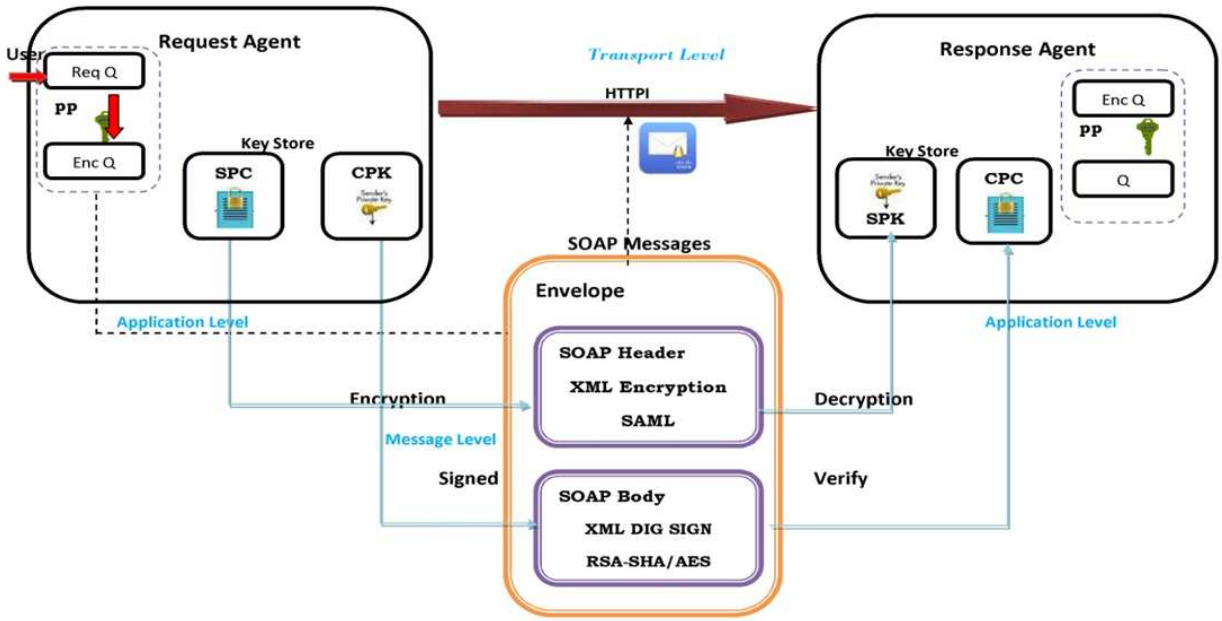

Fig. 2. Request agent to provider agent

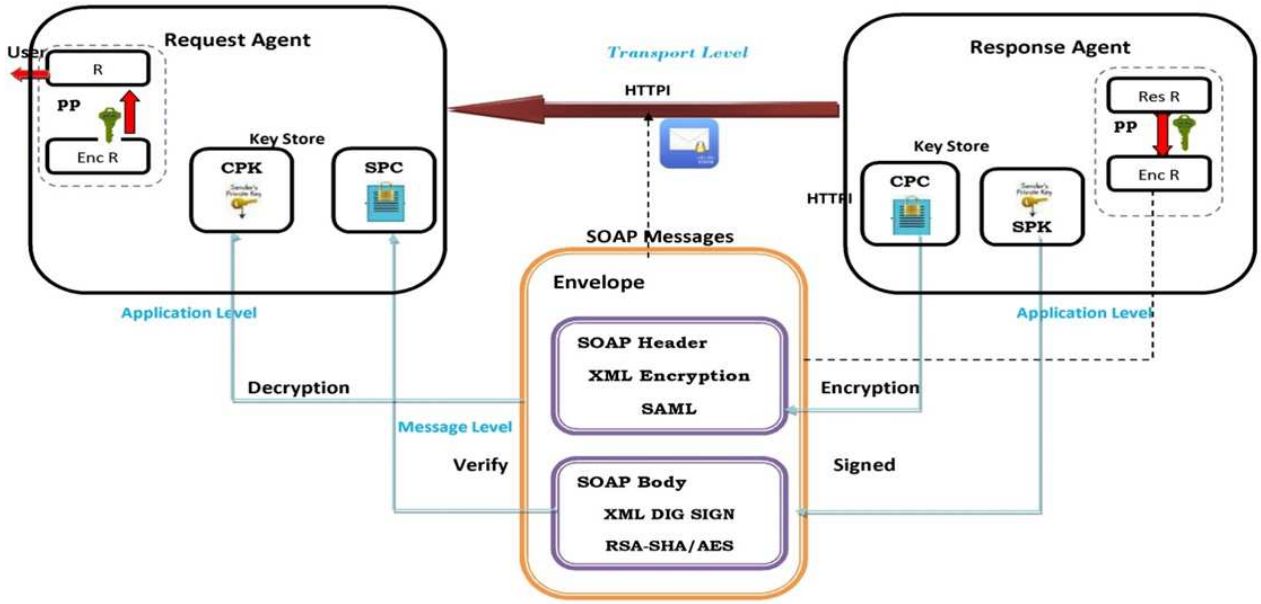

Fig. 3. Response from provider agent to requester agent

In the experimental setup the configuration maintained as Dual Core Processor with 4 GB memory, we conducted the performance test to analyze the HTTPS protocol in various browsers such as Internet Explorer, Mozilla Firefox, Google Chrome for throughput (transaction/seconds), average response time (milliseconds) and response size (KB) and the values are tabulated in **Table 3** and the results are shown as graph in **Fig. 6**.

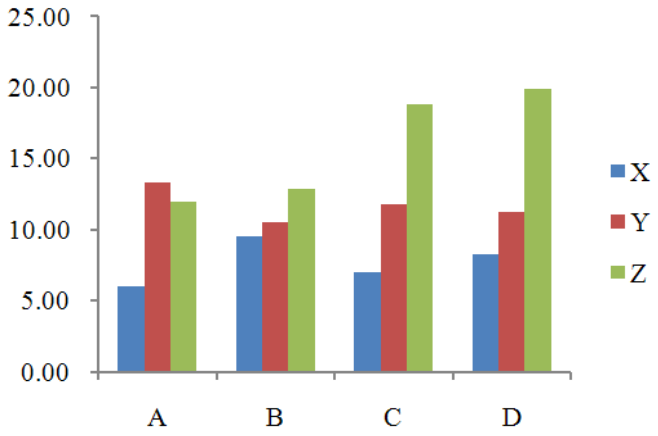

Fig. 4. Graph of web services scenarios Vs average response time, average throughput and reply size X, Y and Z respectively

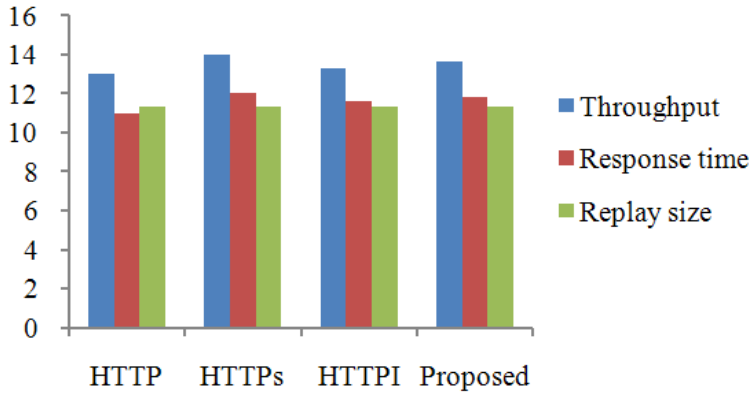

Fig. 5. Protocols HTTP, HTTPs Vs response time, reply size

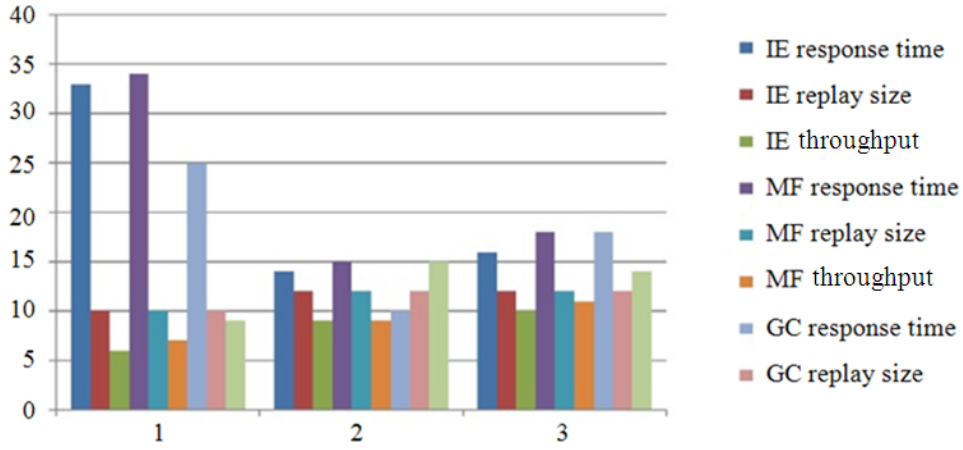

Fig. 6. Comparison of different measurement of HTTPS in different browsers using Dual Core processor 4GB RAM

Table 1. Web services scenarios Vs average response time, average throughput and reply size per request

Scenarios		Average response Time (MS) X	Average throughput (transaction/sec) Y	Reply size per request (Bytes) Z
Non Secure, less Confidentiality	A	6.0	13.33	12.00
Highly Secure ,non supportable for cache proxies, confidentiality	B	9.5	10.50	12.83
Secure ,non Confidentiality, supportable for cache proxies	C	7.0	11.80	18.79
Combination of both highly securable and supportable for cache proxies	D	8.3	11.20	19.84

Table 2. Comparison of various protocols with I3 processor 4GB RAM

I3 Processor	Throughput	Response time	Replay size
HTTP	13.0	11.0	11.3
HTTPs	14.0	12.0	11.3
HTTPI	13.3	11.6	11.3
Proposed	13.6	11.8	11.3

Table 3. Comparison of different measurement of HTTPS in different browsers using Dual Core processor 4GB RAM

Dual core processor 4GB RAM	Internet explorer			Mozilla firefox			Google chrome		
	Response time	Replay size	Throughput	Response time	Replay size	Throughput	Response time	Replay size	Throughput
HTTPS	33	10	6	34	10	7	25	10	9
	14	12	9	15	12	9	10	12	15
	16	12	10	18	12	11	18	12	14

In this study resolution is based on a novel two phases that focuses on achievability of practical composition while the latter deals with execution and next one is use to optimize each stage that can be adopted in service creation.

2. PROPOSED MODEL PRIVACY PRESERVING IN HTTPs AND HTTPi PROTOCOL

In this proposed model, user affords the self encrypted data (privacy preserving) in to requester agent, the requester agent again encrypt that in a particular data, so it can be created two protections, one is self protection and additional one is agent protection. Suppose any hackers are slashes moreover very complicated since the user is doing self encrypting this will give more privacy. The requester agent sends to next level i.e., message level; this level is playing superior responsibility in WS, in this message level i.e., SOAP message level, the header information can be converted in to binary token form in XML encryption using SAML. The encrypted SOAP is integrated to HTTPi in transport level to keep away from the Man in the Middle (MIM) attack.

The **Fig. 2** shows the request agent to provider agent process which is described below:

Step 1: After finished the UDDI registry, the web user want to use a public WS, client used to send an input (request) through requester agent.

Step 2: The (Req Q) requested query encrypting with sender's private key.

Step 3: After encryption the query converted into encrypted query(Enc Q) the step2 and step 3 is dealing with privacy preserving concept

Step 4: The privacy preserving should be any software or application which hide the user data from the hackers, the application level security maintains here and its optional too.

Step 5: The encrypted data will be formed as Soap message while the data transferring the same in httpi protocol.

Step 6: The Soap message encrypted and Signed using receiver's Public Certificate (SPC) and sender's Public Key (CPK) respectively.

Step 7: Xml Encryption: The soap header is bonded with self signed certificate- SAML (binary token).

Step 8: Xml Digital Signature: The soap body content is signed (integrity) using RSA SHA1 algorithm.

Step 9: The digested soap message is transfer to httpi which provide more secure data and avoid man in the middle attack.
Step 10: The encrypted request from the requestor as a Soap message received from the receiver in the other end.
Step 11: The Soap message decrypted and verified using receiver's Private Key (SPK) and (CPC) sender's public certificate respectively.
Step 12: The server/services provider got Encrypted Query from the above process (Enc Q).
Step 13: The Encrypted Queries (Enc Q) have been decrypted using sender's public key and get the resultant query Q.
Step 14: The query Q which provides web user data has been analyzed and responded.

In the response process, the response agent sends back the response with secure manner. The following process has been completed in the responding process. The above steps are explained in diagram the forth coming pages, the main purpose of proposed work is to provide enhanced security in the requester side and responding side. The data is first encrypted with user and then only gives to requester agent then again the requester agent encrypt and the converted to SOAP message. Vice-versa the responding side also the request can be encrypting again encrypted by response agent then converted to SOAP message. Hence these two levels of encryption increase the security level. So Man in the Middle attack is not possible.

3. PROPOSED FORM COMBINATION OF HTTPs AND HTTPi

In this proposed model it has been combined the characteristics of HTTPi and HTTPs. The HTTPs is considered as more secure protocol to transfer data one machine to another especially in WS, but it is not suitable for cache accessibility. Its throughput, response time and reply size are having open difference when compare to other protocols. The HTTPi is considered as very flexible protocol for cache accessibility and security. Though it is very good protocol for transferring WS data it won't provide confidential category security. It is suitable for only social networks like twitter, facebook.

It implies it worth for common accessible data like news or blogs or any public information but not provide any privacy for confidential data. So it has been combined both characteristics of HTTPi and HTTPs to provide the best protocol for WS data transmission. It will give better performance such as excellent throughput, good response time and reply size. Anyway the throughput is depends on the system's configuration like RAM memory's speed Hard disc used, the processor's speed which was measure in Giga hertz and the reply in size is depend on the WS used. The response time was differed from one browser to another browser. That is the response time differed from internet explorer to Google chrome. From the above measures it will be recommended the combination of HTTPi and HTTPs is better protocol for WS to transfer the data:

PP-Privacy Preserving; HTTPi-Hyper Text Transfer Protocol interface; Q-Query; Req Q-Requested Query; Enc Q-Encrypted Query; CPC-Client's Public Certificate; CPK-Client's Private Key; SPC-Server's Public Certificate; SPK-Server's Private Key

4. EXPERIMENTAL RESULTS

Nowadays, more vulnerable attackers are easily hacking the data in WS through installing some tools. This kind of sites how it can be hacked the data and procedure for hacking WS like that Line by Line walk through specified. The WSDL document message will be visible and the attackers can easily hack SOAP message and it request sent by web user.

HTTPs hacking tools release are happen due to security issues. There are actually two vulnerabilities available. The first is that lots of sites do not secure their content via HTTPs past the initial login page. This allows an attacker to take their users cookies and impersonate them on the local network whenever they use the site. The second vulnerability is that many sites that do use https past the login page but do not mark their cookies as 'secure'. This is what allows an attacker to induce their browser to transmit these cookies over unsecured, regular HTTP connections so they can observe them and impersonate the user.

To overcome above mention problem, privacy preserving techniques is proposed which provide highly secure and confidential. In this concept web user encrypt the message before make a request to services provider based on some cryptographic mechanism. While comparing with other encrypted message it is does not have standard and structured format. For example, considered SOAP message which provide structured format due to this vulnerable attackers easily hacking the data using the Line by Line walk through

methodology but in privacy preserving they can hack the encrypt message, it is invisible to observe the original message because in privacy preserving some cryptographic system has been used.

Initially, it has to encrypt the original data in user level itself then after encrypted data sends to requester agent; again the requester agent encrypted before sending to SOAP message. So, when the hackers are attacking in SOAP message level, may be they hack only SOAP level or in requester agent data but Privacy preserving data i.e., user level encrypted data is not possible to hack, because only receiver only can know that key others is not possible to break .The following coding gives explanation about encrypting in the user level privacy preserving:

```
1. public static string Encrypt(string message, string password) {
// Encode message and password
2. byte[]messageBytes=ASCIIEncoding.ASCII.Ge
   tBytes(message);
3. byte[]passwordBytes=ASCIIEncoding.ASCII.G
   etBytes(password)
// Set encryption settings -- Use password for both key
   and init. Vector
4. DESCryptoServiceProvider provider = new
   DESCryptoServiceProvider();
5. ICryptoTransform transform =
   provider.CreateEncryptor(passwordBytes,
   passwordBytes);
6. CryptoStreamMode mode =
   CryptoStreamMode.Write;
// Set up streams and encrypt
7. MemoryStream memStream = new
   MemoryStream();
8. CryptoStreamncryptoStream=new
   ryptoStream(memStream,
9.  transform, mode);
10. cryptoStream.Write(messageBytes,0,messageB
    ytes.Length);
11. cryptoStream.FlushFinalBlock();
// Read the encrypted message from the memory stream
12. byte[]encryptedMessageBytes=newbyte[memSt
    ream.Length];
13. memStream.Position = 0;
14. memStream.Read(encryptedMessageBytes,0,en
    cryptedMessageBytes.Length);
// Encode the encrypted message as base64 string
15. string encryptedMessage =
    Convert.ToBase64String(encryptedMessageBytes);
16. return encryptedMessage; }
```

The SOAP message sends the encrypted data into next level i.e., application level or responding agent. The responding agent decrypted the message and again query sends back to the user encrypted format. So for the three level i.e., Application level, Message Level, Transport level, there are four level of security provided like Authentication, Authorization, Confidentiality, Integrity.

The below coding gives the explanation about decryption of the responding agent:

```
1. public static string Decrypt(string
   encryptedMessage, string password){
 // Convert encrypted message and password to bytes
2. byte[] encryptedMessageBytes =
   Convert.FromBase64String(encryptedMessage);
3. byte[] passwordBytes =
   ASCIIEncoding.ASCII.GetBytes(password);
// Set encryption settings -- Use password for both key
   and init. vector
4. DESCryptoServiceProvider provider = new
   DESCryptoServiceProvider();
5. ICryptoTransform transform =
   provider.CreateDecryptor(passwordBytes,
   passwordBytes);
6. CryptoStreamMode mode =
   CryptoStreamMode.Write;
// Set up streams and decrypt
7. MemoryStream memStream = new
   MemoryStream();
8. CryptoStream cryptoStream = new
   CryptoStream(memStream, transform, mode);
9. cryptoStream.Write(encryptedMessageBytes, 0,
   encryptedMessageBytes.Length);
10. cryptoStream.FlushFinalBlock();
// Read decrypted message from memory stream
11. byte[] decryptedMessageBytes = new
    byte[memStream.Length];
12. memStream.Position = 0;
13. memStream.Read(decryptedMessageBytes, 0,
    decryptedMessageBytes.Length);
// Encode deencrypted binary data to base64 string
14. string message =
    Convert.ToBase64String(decryptedMessageBytes);
15. return message; }
```

In the experimental program it has been calculated average response time, average throughput and reply size per request with different types of scenarios like Non Secure, less Confidentiality (A), Highly Secure, non

supportable for cache proxies, confidentiality (B), Secure, non Confidentiality, supportable for cache proxies (C), Combination of both highly securable and supportable for cache proxies (D). The graph is expressed below.

The below table shows that WS scenarios Vs average response time, average throughput and reply size per request and protocols HTTP, HTTPs Vs Response time, reply Size. It's clearly explains combinations of HTTPs and HTTPi provides more secure and most cache accessibility in the WS.

5. CONCLUSION

WS are extended their service to many of the fields like banking division, business division, educational division. One part WS spreading to all the fields, the second part is security needed to providing web hackers. So day by day have to be enhanced the security level, communication level (Protocol). Hence this study is giving clear idea about new technology i.e., combination of HTTPs and HTTPi technology. So comparing all the protocol the HTTP is providing enormous services, HTTPs are providing security but it is not flexible for example not allowing caches. HTTPi is a flexible but not much more secure like HTTPs for example not providing confidentiality. In the proposed model, it combined the HTTPs's security and HTTPi's flexibility to provide the best WS (transport level). Moreover the user affords the self encrypted data (privacy preserving) in to requester agent, the requester agent again encrypt that in a particular data, it created two protections, one is self protection and additional one is agent protection (application level). Suppose any hackers are slashes moreover very complicated since the user is doing self encrypting this will give more privacy. The requester agent sends the data to next level i.e., message level, in this message level (SOAP), the header information can be self verified using SAML to avoid TPA. The results of this proposal are compared with existing levels and got better throughput and response time. In future to enhance the performance of the WS security combination of HTTPs and HTTPi.

In another experimental setup we maintain the configuration as I3 Processor with 4 GB memory, we conducted the performance test to analyze the HTTP protocol in various browsers such as Internet Explorer, Mozilla Firefox, Google Chrome for throughput (transaction/seconds), average response time (milliseconds) and response size (KB) and the values are tabulated in **Table 4** and the results are shown as graph in **Fig. 7**.

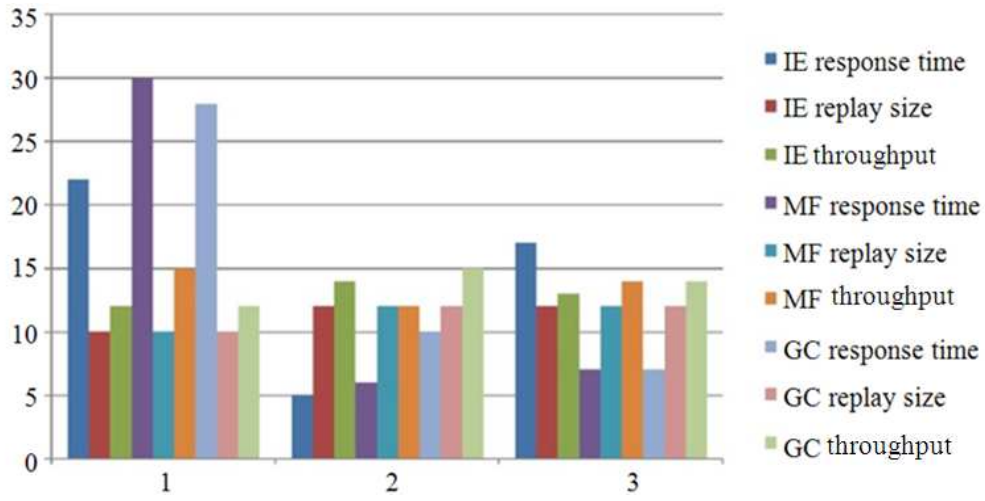

Fig. 7. Comparison of different measurement of HTTP in different browsers using I3 processor 4GB RAM

Table 4. Comparison of different measurement of HTTP in different browsers using I3 processor 4GB RAM

Intel3 processor with 4GBRAM	Internet explorer			Mozilla firefox			Google chrome		
	Response time	Replay size	Throughput	Response time	Replay size	Throughput	Response time	Replay size	Throughput
HTTP	22	10	12	88	10	15	28	10	12
	5	12	14	6	12	12	10	12	15
	17	12	13	7	12	14	7	12	14

6. REFERENCES

Choudhary, P. and R.A. Nirmal, 2013. HTTPi based web service security over SOAP. Int. J. Netw. Sec. Applic., 5: 55-66. DOI: 10.5121/ijnsa.2013.5306

Hao, Z., S. Zhong and N. Yu, 2011. A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability. IEEE Trans. Knowl. Data Eng., 23: 1432-1437. DOI: 10.1109/TKDE.2011.62

Jian, Z., 2011. A Web services-based security model for digital watermarking. Proceedings of the International Conference on Multimedia Technology, Jul. 26-28, IEEE Xplore Press, Hangzhou, pp: 4805-4808. DOI: 10.1109/ICMT.2011.6002056

Rathore, M. and U. Suman, 2011. A quality of service broker based process model for dynamic web service composition. J. Comput. Sci., 7: 1267-1274. DOI: 10.3844/jcssp.2011.1267.1274

Xu, R., B. Ji, B. Zhang and P. Nie, 2013. Research on dynamic business composition based on web service proxies. Simulat. Modell. Pract. Theory, 37: 43-55. DOI: 10.1016/j.simpat.2013.05.008

Zhang, J., 2011. A web services-based security model for digital watermarking. Proceedings of the International Conference on Multimedia Technology, Jul. 26-28, IEEE Xplore Press, Hangzhou, pp: 4805-4808. DOI: 10.1109/ICMT.2011.6002056