File Transfer Using MTOM Technique Using Web Services And Performance Evaluation Of MTOM And SwA

Pratiksha J.Deshmukh
M.E. Student,Computer Engineering Department
Shree L.R.Tiwari College Of Engineering,Mira Road,
Mumbai,India

Prof.Amarja Adgoankar
Assistant Professor,HOD,Computer Engineering Department,K.C.College of Engineering,Thane,Mumbai,India

Anil Chaturvedi
Assistant Professor,Computer Engineering Department
Shree L.R.Tiwari College Of Engineering,Mira Road,
Mumbai,India

Abstract— **Web Service is a technology that is based on the Service Oriented Architecture.It enables communication between applications through the Internet.Using Web Services,it is possible to send any type of information in any form of encryption. Many different techniques are used for sending binary files as an attachment to SOAP messages.The W3C recommendation MTOM(Message Transmission Optimization Mechanism) is the standard for transferring binary files in SOAP attachment-the efficient technique for transferring binary data without breaking of XML infoset.This Paper presents MTOM Web Service technique and performance evaluation study with two techniques for Web Service attachments: MTOM and SwA. A testing environment will be configured to verify the influence of the network and the size of files with respect to Time.Performance of MTOM is calculated depending upon chunk size and threads with respect to time.The problem of DIME(Direct Internet Message Encapsulation)is that the content of binary form is sent outside of the the SOAP Envelope of XML message.Because of that DIME attachment may not be secure.But in MTOM, web service directly handles data encoded in web service message.The implementation of MTOM will give approximately 10-15% faster result than DIME.**

Index Terms—**DIME,MTOM,SOAP,SwA,XML infoset**

I. INTRODUCTION

Data transfer is the fundamental building block for distributed applications which are implemented using Web Services. Data transfer using SOAPmessage requires XML documents with embedded data. XML represents semi-structured data inside text-based documents. But this textual data are not applicable for all domains. So,we required technology that will include binary data into XML documents[2].

Message passing is used for communication in Service Oriented Architecture.Web service is the most commonly used technology to implementSOA.In Web services, SOAP messages encapsulate data attachments in a different ways then campared to traditional services.Howerever,Developer has to take into account various factors like file type,network traffic and file size that is attached to SOAP message.[1]

II. BASIC CONCEPT

SOAP(Simple Object Access Proctocol) is a protocol for exchanging information in structured form in web services. For its message format it uses XML info set.DIME(Direct Internet Message Encapsulation) is an internet standard for streaming of binary and other encapsulated data on internet.DIME was replaced by MTOM and SwA.SwA is SOAP with Attachments or also known as MIME for web services.SwA is a mechanism for using SOAP and MIME facilities for transmission of files using web services.Multipurpose Internet Mail Extension(MIME) is an extension of internet e-mail protocol that is used to exchange many kinds of data files over internet.MTOM is a W3C Recommendation for transferring binary Data in SOAP messages.MTOM is used with XML-Binary Optimized Packaging.[2]

ISSN: 2278 – 1323

I. FEATURES IN FILE TRANSFER

1. If we have a very large file that we need to send across a network using web service, you send it as an array of bytes as a parameter to a web service call, which is all sent to the IIS web server as a single request. This is bad if the size of file is beyond the configured Max-Length or if the request causes an IIS timeout. The sender has a file(s) to send to the receiver. But, size of the file may be too large for transmitting it as a single entity. So, the sender's buffer will include the parts of the file. These parts of the file are called 'chunks' and each chunk can be sent individually. The number of chunks is variable for different file types. This number is decided by the chunk size and the size in which the original file will be broken down. These chunks are then transmitted through Web Service. The receiver then receives these copies multiple copies of the same chunk are received; it will keep the first copy and discard the duplicate copies. We will implement SHA (Secure Hashing Algorithm) which will be used by the web service. This hashing will be done on the client and the server to verify that the file received is identical to the file sent.

2. A progress bar will also be implemented for file transfer, giving a success message and time required for the transfer as soon as the file is transferred successfully. If any error occurs, an error message will pop up.

3. Pause and Resume functionalities while the transfer is in progress will also be implemented.

II. FILE TRANSFER PROCESS

Figure 1. shows the file transfer mechanism. Here, the sender has a file to send to the receiver. But, size of the file may be too large for transmitting it as a single entity. So, the sender's buffer will include the parts of the file. These parts of the file are called 'chunks' and each chunk can be sent individually. The number of chunks is variable for different file types. This number is decided by the chunk size and the size in which the original file will be broken down. But, breaking a file will create inconsistency and will be difficult to retain.

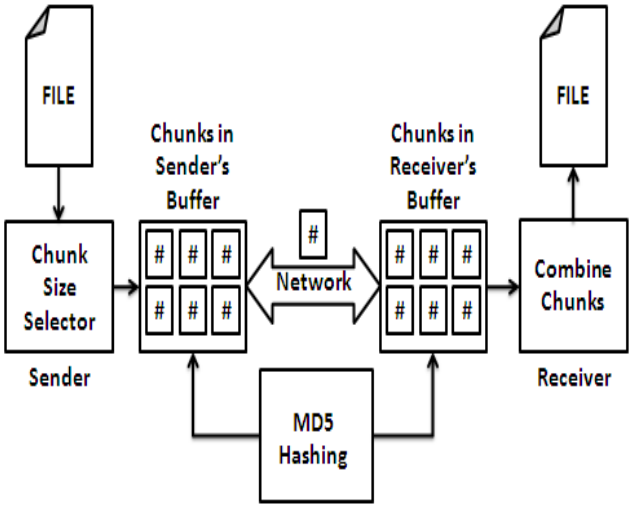


Figure 1: Process of File Transfer

The solution for this is to use MD5 Hashing Algorithm, which attaches a hash value to the file. A hash is created by taking a string of any length and encoding it to a 128-bit value. This value forms a fingerprint for that file. Encoding the same string will always result in the same 128-bit hash value. MD5 hashes are basically used with smaller strings. For example, storing passwords, credit card numbers etc.

MD5 hashes are used to ensure the integrity of files. As the algorithm always produces the same output for the same input, users can compare the newly created hash value with the hash value of the source file to check that it is intact and unmodified. An MD5 hash is NOT any encryption mechanism. It is simply a fingerprint of the given input. However, it is a one-way transaction and it is almost impossible to retrieve the original string from a hash value.[1]

The input message is broken up into chunks of 512-bit blocks. The message is padded so that its length is divisible by 512. The padding works as follows:

i. First a single bit, 1, is appended to the end of the message.

ii. Then it is followed by as many zeros as are required to bring the length of the message up to 64 bits less than a multiple of 512.

ISSN: 2278 – 1323

iii. The remaining bits are filled up with 64 bits representing the length of the original message, modulo 264.

The main MD5 algorithm operates on a 128-bit state, divided into four 32-bit words. These are initialized to certain fixed constants. The main algorithm then operates on each 512-bit message block in turn. The processing of a message block consists of four stages. Each stage is composed of 16 similar operations based on a non-linear function, modular addition, and left rotation.

Each chunk will have unique hash value but signature will be same if they are part of same file. These chunks are then transmitted through Web Service using MTOM as shown in Figure 2. A file to be sent is selected for transfer. The size of the file may be too large for sending at a particular time because the bandwidth available may be too low. This file is then passed through the chunk size selector, where the file is divided into smaller chunks suitable for transmission. The size of the chunk can be decided by the sender or if the sender does not select the size may vary depending upon the type of data, amount of data and availability of the network resources.

These chunks are then buffered at sender side and hashing is performed. Now the chunks with hash value are sent through the network. When receiver receives these chunks, it buffers then in its own buffer. It then checks for the hash value of received chunks. If all chunks are received correctly, they are combined to form the complete file. If hash value does not match error is generated.

A. Message Transmission Optimization Mechanism

SOAP Message Transmission Optimization Mechanism (MTOM), is a W3C Recommendation designed for optimizing the electronic transmission of attachments. Through electronic transmission of documents, corporations can realize significant cost savings and better service levels by eliminating the use of postal mail. Paper-based manual tasks can be replaced with simple and efficient electronic processes where binary data can be transmitted between organizations through standards such as MTOM.

MTOM provides an elegant mechanism of efficiently transmitting binary data, such as images, PDF files, MS Word documents, between systems. The Figure a below shows the steps involved in transmitting data between a Consumer and Producer using MTOM .[1]

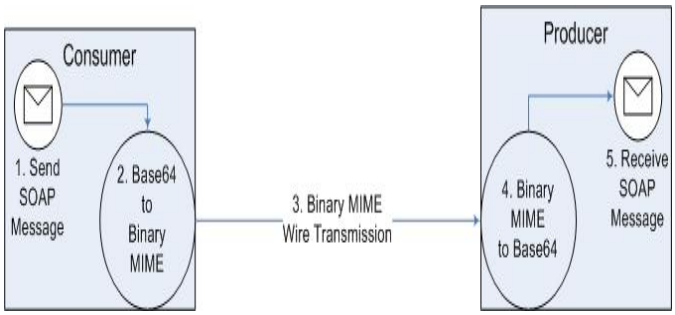


Figure 2: Transmitting data using MTOM

B. MTOM Chunks

MTOM Chunks are the smallest part of the file that can be individually transferred. This chunk gets assigned by hashing value and a signature. Each chunk will have different hash values depending upon the content of the chunk but the signature will be same for chunks belonging to the same file.

Each chunk will generate XML XOP which will be useful while regenerating the chunk at the receiver side. When XML XOP and hash value gives accurate result, the chunk is accepted otherwise it gets discarded. If any problem occurs in regeneration, the file may not be consistent and its integrity is affected thus making the file unusable.[1]

C. MD5 Hashing

MD5 Hashing Algorithm is used to produce a message digest for a given message. Essentially, this is a 128-bit number that represents the message. A hash function is an algorithm that takes an arbitrary block of data and returns a fixed-size bit string, the (cryptographic) hash value, such that an (accidental or intentional) change to the data will

ISSN: 2278 – 1323

(with very high probability) change the hash value. The data to be encoded is often called the "message," and the hash value is sometimes called the message digest or simply digests. A message digest can serve as a finger print for a file or other source of data. Hence, no two messages would ever share the same message digest.It shows that how the digest value changes even if one character in the string is different in two strings.

The ideal secure hash function has four main or significant properties:

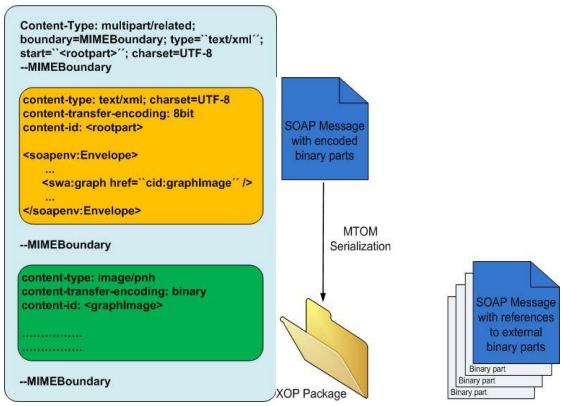
- It is easy to compute the hash value for any given message
- It is infeasible to generate a message that has a given hash
- It is infeasible to modify a message without changing the hash
- It is infeasible to find two different messages with the same hash

Hash functions have many information security applications, notably in digital signatures, message authentication codes (MACs), and other forms of authentication. They can also be used as ordinary hash functions, to index data in hash tables, for fingerprinting, to detect duplicate data or uniquely identify files, and as checksums to detect accidental data corruption. Indeed, in information security contexts, cryptographic hash values are sometimes called (digital) fingerprints, checksums, or just hash values, even though all these terms stand for functions with rather different properties and purposes.[1]

The goal of the project is to design and implement a middleware services for data transfer on network. This middleware will provide the services for data transfer for large files on network. While transferring large file over network using web service, an array of bytes as a parameter is sent to the IIS server as a single request.

This transfer is inappropriate if the size of the file is beyond configured length or if the request causes IIS Timeout. So, a mechanism will be implemented to break this large file into small chunks and send these chunks individually.

Secure Hashing is used so that all the chunks can be regrouped at receiver side and file will be successfully received without any issues. Another drawback of transferring large file is that is you get feedback until it is either completely transferred or failed.[2]



(a) SwA[1] (b) MTOM[2]

Figure 3:SwA and MTOM

III. CONCLUSION

It is expected that the file transfer speed will be 10-15% faster for uploading and downloading by using MTOM web service technique than DIME.This is expected because in MTOM there will be file divided into chunks.

.

REFERENCES

[1]. Analysis Based on Chunk Size and Threads for File Transfer Using Message Transmission Optimization Mechanism(MTOM)

[2]. A Performance Evaluation Study for Web Services Attachments (IEEE International Conference on Web Services, 2009)

[3]. Server-side Streaming Processing of Secured MTOM Attachments (Eighth IEEE European Conference on Web Services, 2010)

[4]. "S.-J. Cha, Y.-Y. Hwang, Y.-S. Chang, K.-O. Kim, and K.- C. Lee. The Performance Evaluations and Enhancements of GIS Web Services. In *MUE '07: Proceedings of the 2007 International Conference on Multimedia and Ubiquitous Engineering*, pages 668–673, Washington, DC, USA, 2007. IEEE Computer Society. [5]. "SOAP Message Transmission Optimization Mechanism", http://www.w3.org/TR/soap12-mtom/

[6]. "Sending files in chunks with MTOM Web Services", http://tim.mackey.ie/SendingFilesInChunksWithMTOMWebServicesAndNET20.aspx

ISSN: 2278 – 1323

[7]. M. Gudgin, M. Hadley, N. Mendelsohn, J.-J. Moreau, and H. F. Nielsen, "SOAP Version 1.2 Part 1: Messaging Framework," W3C Recommendation, 2003.

[8]. Nils Gruschka, Luigi Lo Iacono, "Server-side Streaming Processing of Secured MTOM Attachments", IEEE Computer Society, 2010, Page 11 – 18.

[9]. J. C. Estrella, F. J. Monaco, R. H. C. Santana, and M. J. Santana. Real Time Compression of Soap Messages in a Soa Environment. *SIGDOC'08: Proceedings of the 26th Annual ACM International Conference on Design of Communication*, pages 163–168, 2008.

[10] . R. R. Kodali. What is service-oriented architecture?, 2005. Available at: http://www.javaworld.com/javaworld/jw-06-2005/jw-0613-soa.html.

[11]. Julio Cezar Estrella, Andr´e Takeshi Endo, Rubens Kenji T. Toyohara, Regina H. C. Santana, Marcos J. Santana, Sarita Mazzini Bruschi, "A Performance Evaluation Study for Web Services Attachments", IEEE Computer Society, 2009, Page 799 – 806.

ISSN: 2278 – 1323