Webology (ISSN: 1735-188X)
Volume 18, Number 6, 2021

Effect Of Blob And String Data Type On Time And Space Consumption In Recognition Based Image Authentication Scheme

Zeeshan I. Khan[1] , Vijaya K. Shandilya[2]

[1] Research Scholar, Sipna COET, Amravati., ORCID ID: orcid.org/ 0000-0002-5128-9444

[2]Professor, Sipna COET, Amravati.

Abstract

Authentication is the process of recognizing a user's identity on any system or network. The authentication process always runs at the start of the application to authorize the user. Recognition based image authentication scheme (RBIAS) is one of popular method of image authentication scheme in which user can select its password in the form of images to remember it easily. A binary large object (BLOB) is a collection of binary data stored as a single entity. Blob data type is typically used to store images, audio or other multimedia objects. In this paper, we have given the comparison of time and space complexity between blob data type and string data type at the time of image storage and verification in recognition based image authentication scheme. The results are analyzed by implementing both (blob and string) data type techniques and given a productive output showing which data type should be used while storing images in image authentication scheme.

Keywords: RBIAS, Images, Blob, String.

Introduction

Now a days, every system (PC or Mobile Phone) needs a strong authentication method to check the validity of the user. Recognition based image authentication scheme (RBIAS) is one of popular method of image authentication scheme in which user can select its password in the form of images to remember it easily. This Technique allows the user to select the images, graphics, from the list of given examples stored on the server and recognize it at the time of authentication. For Example: Image selection from a given set of images is used as a captcha in which user has to select an image as per the given instruction or a question given on the screen. It is an example of a recognition-based authentication scheme [1].

In Recognition based image authentication scheme (RBIAS), Blob data type is commonly used. A binary large object (BLOB) is a collection of binary data stored as a single entity. Blobs are typically images, audio or other multimedia objects, though sometimes binary executable code is stored as a blob. They can exist as persistent values inside some databases, or exist at runtime as program variables in some languages [2].

Literature review

Reshma and G. Shivaprasad [1] have given an heirarcy of different authentication schemes given in the world as given below,

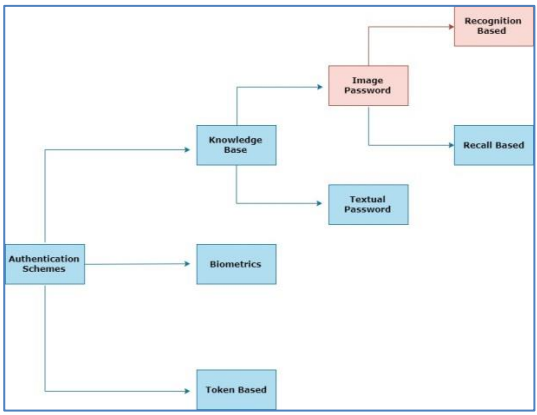


Figure 1. Type of Authentication Schemes

Wayne Jansen, Serban Gavrila, Vlad Korolev, Rick Ayers and Ryan Swanstrom [3] proposed an authentication scheme in which image thumbnails are given to the user for image selection. When the user selects an image, a randomly generated text (attached with the image cell) is stored in the database as a password. This technique saves the memory at the front end and back ends both. At the front end, due to image thumbnail, memory is saved and at the back end, due to randomly generated text, memory is saved [3].

Ting-Yi Changa, Cheng-Jung Tsaib and Jyun-Hao Lina [4] proposed a graphical authentication scheme for touch screen devices. In this system, users have to select images by click on the touch screen. It stores the time interval and pressure at the time of image selection which acts as a password [4].

Vikas K. Kolhekar and Milindkumar B. Vaidya [5] proposed a click and session-based scheme for smartphones and the web. This scheme contains 2 methods using the concept of captcha as a graphical password [5].

Amol Bhand, Vaibhav Desale, Swati Shirke and Suvarna Pansambal (Shirke) [6] proposed a system in which user has to upload any image of his/her choice or can select an image from the existing database. After selecting the image, the user has to select some cued click points on that image. Based on the RGB values of the selected image, a textual password is generated and sent on the email of the user [6].

Swaleha Saeed and M. Sarosh Umar [7] implemented an authentication scheme in which the user has to select an image from the given set of 16 images by entering a three-digit code written on the image. At the time of password verification, the images are displayed on the screen and below the images the balls of different colors are also given [7].

Blobs were originally just big amorphous chunks of data invented by Jim Starkey at DEC, who describes them as "the thing that ate Cincinnati, Cleveland, or whatever" from "the 1958 Steve McQueen movie", referring to The Blob. Later, Terry McKiever, a marketing person for Apollo, felt that it needed to be an acronym and invented the backronym Basic Large Object. Then Informix invented an alternative backronym, Binary Large Object [8].

The Blob object represents a blob, which is a file-like object of immutable, raw data; they can be read as text or binary data, or converted into a Readable Stream so its methods can be used for processing the data.
Blobs can represent data that isn't necessarily in a JavaScript-native format. The File interface is based on Blob, inheriting blob functionality and expanding it to support files on the user's system [9].

Sachin Kaja and Divya Gupta [10] implemented an authentication scheme using persuasive cued click points in which an image is

displayed on the screen and the user has to click at any portion of an image and set that portion as a password. In this authentication scheme, there are various positions in a single image a user can choose. For example, consider an image of a bedroom having different objects like bed, table, chair, TV; clock, etc. and the user can select any of the given objects as a password by clicking on it [10].

R. Sudha and M. Shanmuganathan [11] implemented an authentication scheme in which a user has to upload an image of his choice given by the server and then the server breaks that image in the grid of 7 × 11. After breaking those images in pieces, all the pieces are displayed to the user and then the user has to select any one of those pieces as a password [11].

Shums Tabrez and Jagadeesh Sai D [12] proposed a scheme in which the combination of OTP (one-time password) and the row, column number from an image gives a password.
The whole process is done by giving an image containing gridlines on the screen [12].

Research methodology

The implementation of the recognition based image authentication scheme is done in JavaScript Language by taking few collections of Images. From these images, user has to select one or more images as a password. When a user selects multiple images as a password, that selected images should be stored inside the memory. At the time of password verification, the panel of many images will be given to the user and user has to recognize its correct images from the given set. After Selection, that selected images will be processed and compares with the stored images inside the memory. If the Selected images are same as stored images, the password is correct else the authentication

fails.

This process is implemented by using two different data types in JavaScript,

Method 1: Using Blob Data type

The images will be stored in the memory using blob data type which uses array buffer to store all the values of image in binary format.

Method 2: Using String Data type

The images will be converted into alphabets, numbers and symbols at front end and that combination of alphabets-string and numbers will be stored in the memory.

System Design

Methodology I: Using Blob Data type

In the below algorithm of RBIAS using blob data type, we are fetching an image through its URL and store that image in a variable of blob data type, converting it into array buffer and comparing it with other images to check time and space consumption of existing methodology.

Image_Password (Storage & Verification)

Step 1: Fetch an image through Image URL and stores it in a variable srcimage.

Step 2: Call Image_to_Blob (srcimage) and accept blob1 & blob1 size of an Image 1.

Step 3: Call Image_to_Blob (srcimage) and accept blob2 & blob2 size of an Image 2.

Step 4: Call Blob_to_ArrayBuffer (blob1) and accept buffer1 of blob l image.

Step 5: Call Blob_to_ArrayBuffer (blob2) and accept buffer2 of blob2 image.

Step 6: Call Compare _Image_Buffers (buffer1, buffer2) and accept

the value in the variable result.
Step 7: if (result ==true)
Write (“Authentication Successful”)
else
Write (“Authentication Failed”)

Methodology II: Using String Data type

The below algorithm of RBIAS is designed to enhance the performance of RBIAS in terms of time and space complexities. In this algorithm, the images selected from panel 1 are converted into alphabets (A to Z), the images selected from panel 2 are converted into numbers (0 to 9), and images selected from panel 3 are converted into symbols (15 Sample Symbols taken like ! @ # $ % ^ & * etc.).

Enhanced_Image_Password (Storage & Verification)

Step 1: Set the image ids (A to Z) of 26 Images from Panel 1.
Step 2: Set the image ids (0 to 9) of 10 Images from Panel 2.
Step 3: Set the image ids (Any 15 Symbols) of 15 Images from Panel 3.
Step 4: Call Image_to_Alphabets (Panel1) and accept the result in the variable pwd1.
Step 5: Call Image_to_Numbers (Panel2) and accept the result in the variable pwd2.
Step 6: Call Image_to_Symbols (Panel3) and accept the result in the variable pwd3.
Step 7: Concatenate pwd1, pwd2, pwd3 in a single variable password and stored it in the memory. i.e. password= pwd1 & pwd2 & pwd3.
Step 8: Select the images from Panel 1 for signin.
Step 9: Select the images from Panel 2 for signin.
Step 10: Select the images from Panel 3 for signin.
Step 11: Call Image_to_Alphabets (Panel1) and accept the result in the variable epwd1.

Step 12: Call Image_to_Numbers (Panel2) and accept the result in the variable epwd2.
Step 13: Call Image_to_Symbols (Panel3) and accept the result in the variable epwd3.
Step 14: Concatenate epwd1, epwd2, epwd3 in a single variable epassword and stored it in the memory. i.e. epassword= epwd1 & epwd2 & epwd3.
Step 15: if (password ==epassword)

Write (“Authentication Successful”)

else

Write (“Authentication Failed”)

System Implementation

Methodology I: Using Blob Data type

Step 1: Image Selection Panel made in JavaScript

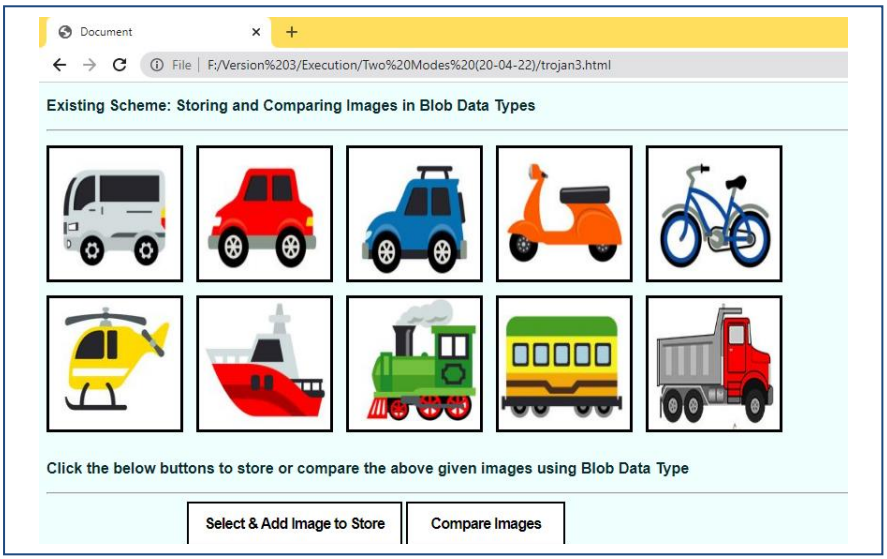


Figure 2. Image Selection Panel

Step 2: Two Images Storage and Space Consumed

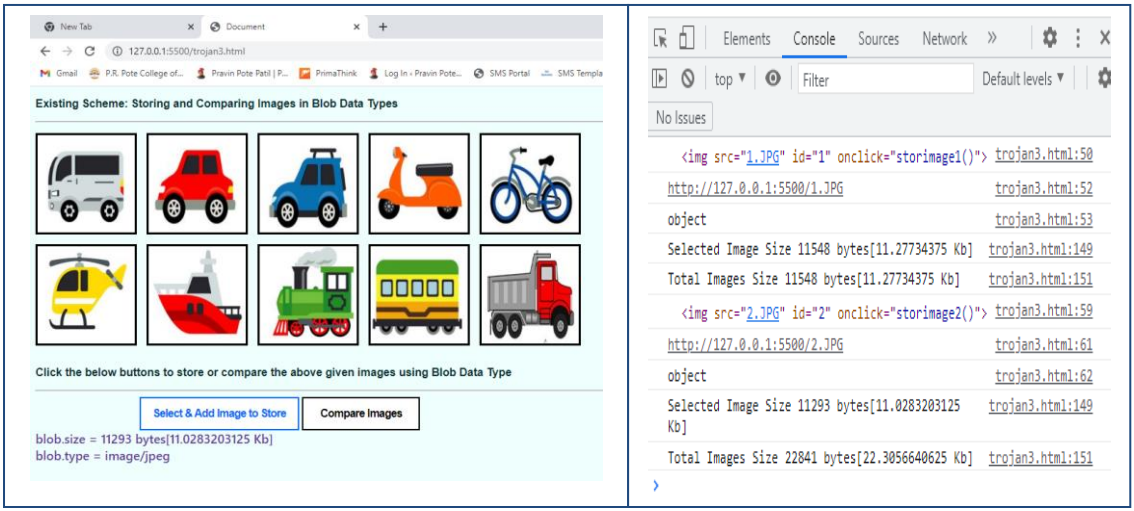


Figure 3. Images Stored (Left Side) & Space Calculated (Right Side)

Step 3: Two Images Compared and Time Consumed (476978 ms)

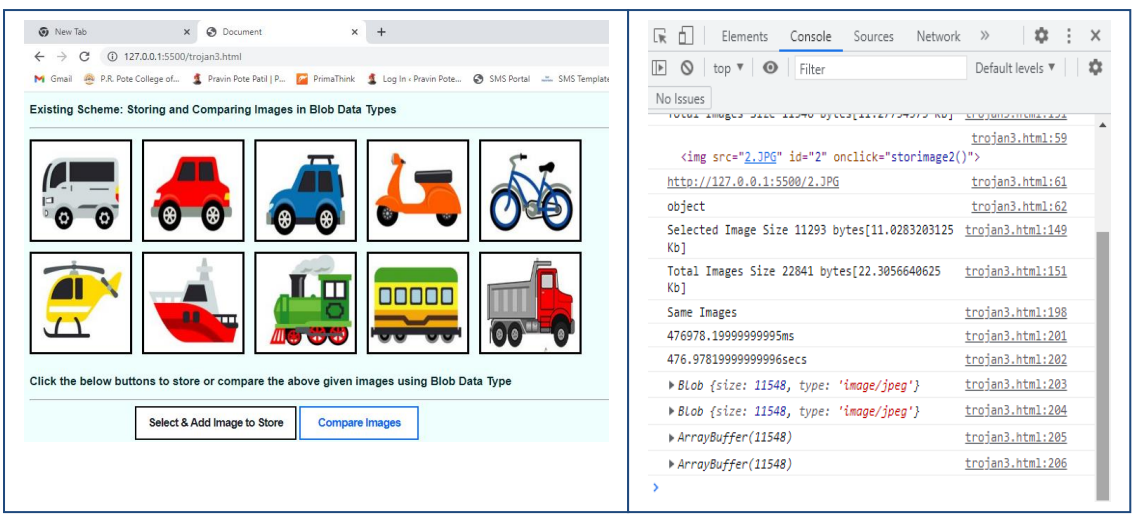


Figure 4. Images Comparison (Left Side) & Time Calculated (Right Side)

Methodology II: Using String Data type

Step 1: Image Selection Panel I and stored alphabets (CD) for selected images 3 & 4

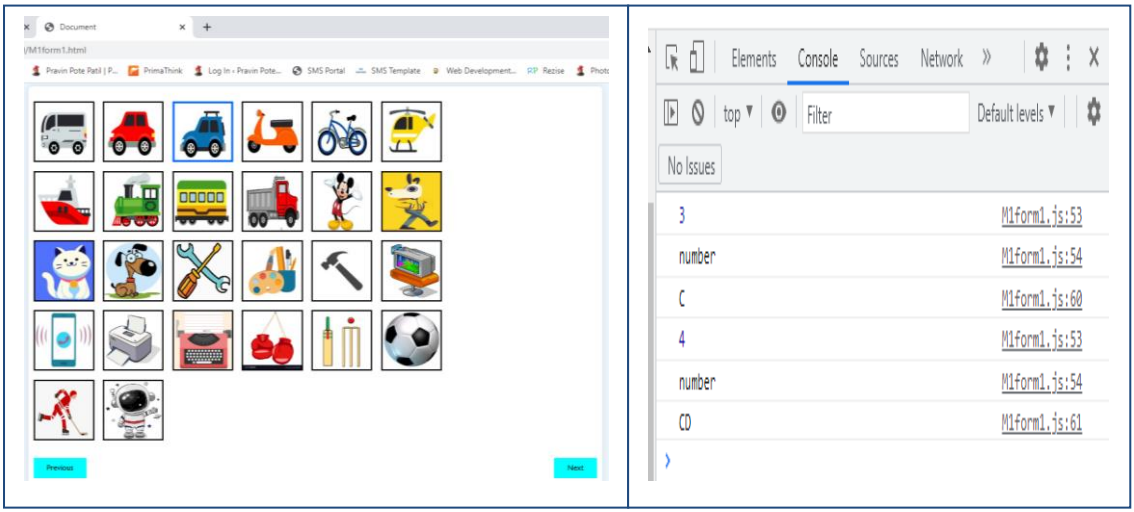


Figure 5. Images Selection Panel I

Step 2: Image Selection Panel II and stored numbers (01) for selected images 1 & 2

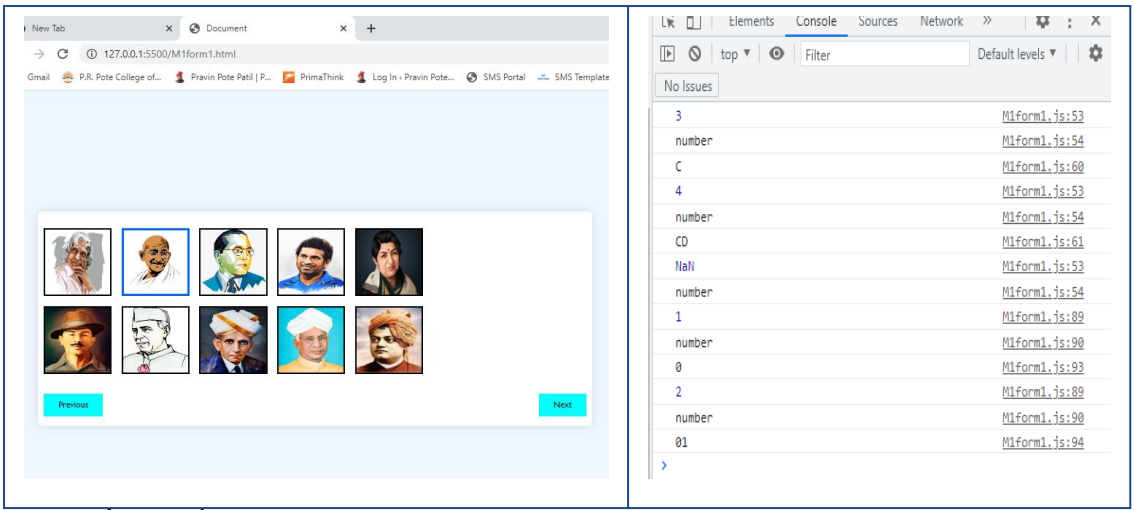


image 4

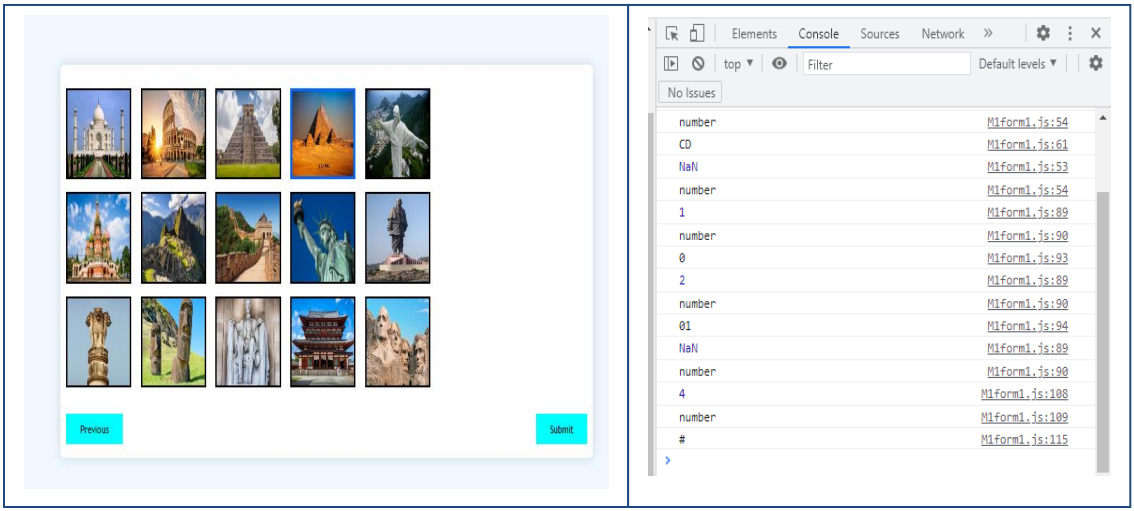
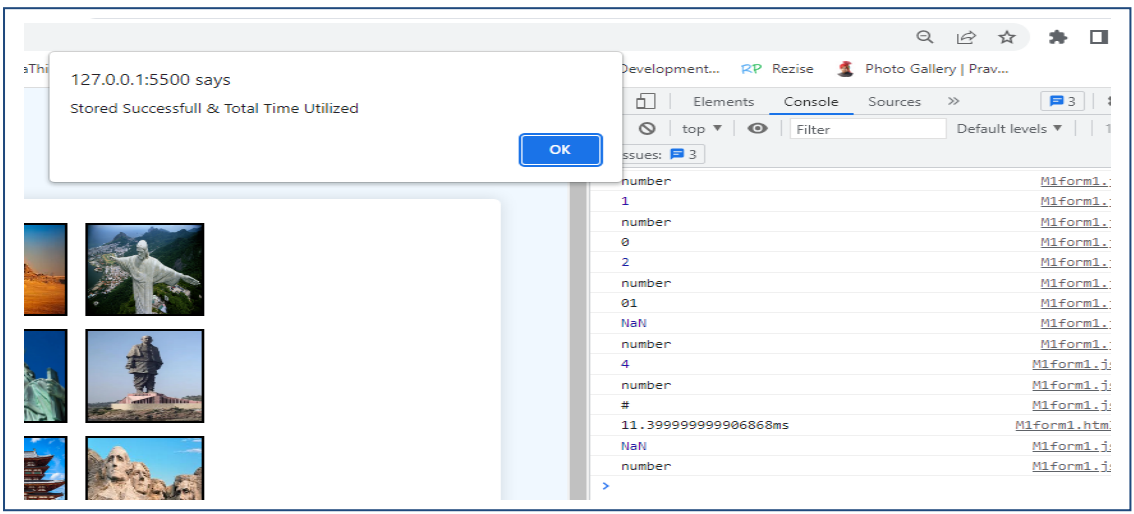


Figure 7. Images Selection Panel III

Step 4: Total Time Required for Images Stored (11.399 ms)



Step 5: Total Time Required for Images Comparison (7.09 ms)

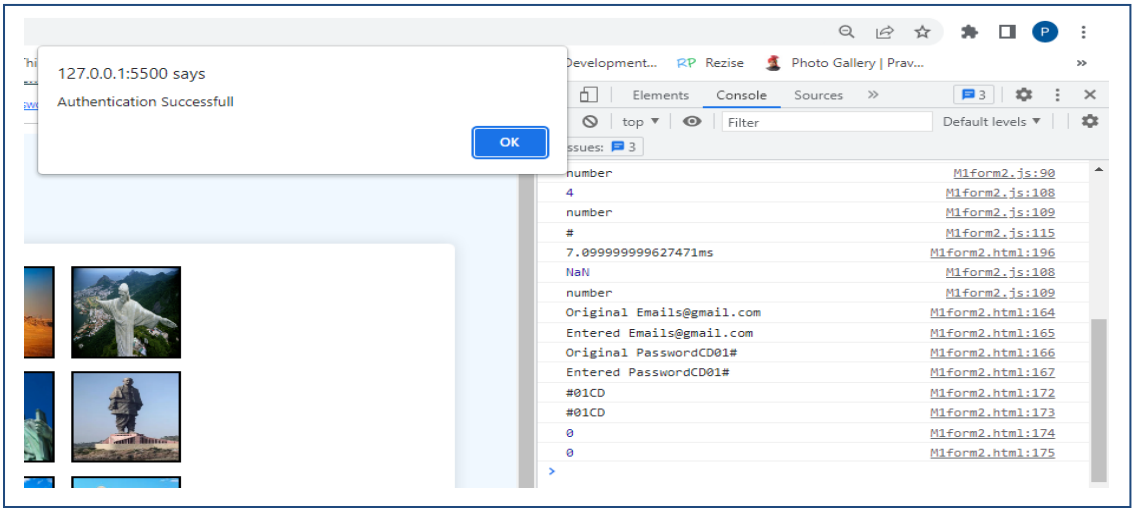


Figure 9. Images Comparison (Left Side) & Time Calculated (Right Side)

Evaluation of Implemented Methods

In JavaScript performance.now() method is used to check the performance of our code. By Using Method 1 (blob data type) algorithm, execution time of image comparison is calculated. By Using Method 2 (String data type) algorithm, execution time of image comparison is calculated.

Results

Space Consumption by Method 1 & Method 2

We have taken the example of Three Different Images and it is stored in memory using blob data type.

Sr. No.	Image Name	Actual Image Size	Blob Size when Stored as a Password

1	1.jpg	11548 bytes	11548 bytes
2	2.jpg	11293 bytes	11293 bytes
3	3.jpg	11923 bytes	11923 bytes

Table 1. Space Consumption by Method I

Again, we have used the above images and it is stored in memory using String data type.

Sr. No.	Image Name	Actual Image Size	String Size when Stored as a Password
1	1.jpg	11548 bytes	1 byte
2	2.jpg	11293 bytes	1 byte
3	3.jpg	11923 bytes	1 byte

Table 2. Space Consumption by Method II

Time Consumption by Method 1 & Method 2

We have taken the example of Different Images and calculated the time by using blob data type.

Sr. No	Comparison Between		Processing Time Calculated (Milliseconds & Seconds)
	First Image	Second Image	
1	1.jpg	1.jpg	3894.20 ms
2		2.jpg	1421.59 ms
3		3.jpg	2685.69 ms

Table 3. Time Consumption by Method I

Again we taken the same example of Different Images and calculated the time by using string data type.

Sr. No	Comparison Between		Processing Time Calculated (Milliseconds & Seconds)
	First Image	Second Image	

1	1.jpg	1.jpg	6.10 ms
2		2.jpg	5.80 ms
3		3.jpg	6 ms

Table 4. Time Consumption by Method II

Conclusions

The Recognition based image authentication scheme has been implemented using two different data types. In first Method, blob data type is used and then time and space consumption is calculated. In second Method, string data type is used and then time and space consumption is calculated. From the Analysis of both the methods (1: blob data type & 2: string data type), we have found that the memory is stored at larger extent when string data type is stored. Similarly, processing time is also decreased in the method 2 (using string data type).

References

1. Reshma and G. Shivaprasad, "Research and Development of User Authentication using Graphical Passwords: A Prospective Methodology," International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 8, no. 9S3, pp. 385-390, July 2019.
2. Wikipedia. Wikipedia Web site. [Online]. https://en.wikipedia.org/wiki/Binary_large_object#:~:text=A%20binary%20large%20object%20(BLOB,is%20stored%20as%20a%20blob
3. Wayne Jansen , Serban Gavrila, Vlad Korolev, Rick Ayers, and Ryan Swanstrom, "Picture Password: A Visual Login Technique for Mobile Devices," National Insitute of Standards and Technology, Gaithersburg, USA, NISTIR 7030, 2003.

4. Ting-Yi Chang, Cheng-Jung Tsai, and Jyun-Hao Lin, "A graphical-based password keystroke dynamic authentication system for touch screen handheld mobile devices," The Journal of Systems and Software, Science Direct, vol. 85, no. 5, pp. 1157-1165, January 2012.
5. Vikas K. Kolekar and Milindkumar B. Vaidya, "Click and Session Based- Captcha as Graphical Password Authentication Schemes for Smart Phone and Web," in International Conference on Information Processing (ICIP), IEEE, Pune, 2015, pp. 669-674.
6. Amol Bhand, Vaibhav Desale, Swati Shirke, and Suvarna Pansambal (Shirke), "Enhancement of Password Authentication System Using Graphical Images," in International Conference on Information Processing (ICIP), IEEE, Pune, 2015, pp. 217-219.
7. Swaleha Saeed and M. Sarosh Umar, "A Hybrid Graphical User Authentication Scheme," in Communication, Control and Intelligent Systems (CCIS), IEEE, Mathura, 2015, pp. 411-415.
8. James Starkey. Web Archive Web site. [Online]. https://web.archive.org/web/20110723065224/http://www.cvalde.net/misc/blob_true_history.htm
9. MDN. Mozilla Web site. [Online]. https://developer.mozilla.org/en-US/docs/Web/API/Blob
10. Sachin Kaja and Divya Gupta, "Graphical Password Scheme using Persuasive Cued Click Points," in International Conference On Smart Technology for Smart Nation, IEEE, Bangalore, 2017, pp. 639-643.
11. R. Sudha and M. Shanmuganathan, "An Improved Graphical Authentication System to Resist the Shoulder Surfing Attack," in International Conference on Technical Advancements in

Computers and Communications, IEEE, Melmaurvathur, 2017, pp. 53-55.

12. Shums Tabrez and Jagdeesh D Sai, "Pass-Matrix authentication," in International Conference on Intelligent Computing and Control Systems, IEEE, Madurai, 2017, pp. 776-781.