IJIRAE:: International Journal of Innovative Research in Advanced Engineering ISSN: 2349-2163
Volume 11, Issue 12, December 2024 https://www.ijirae.com/archives

User Authentication System Using Python

Pawan Mishra
Dept. of Information Technology,
Greater Noida Institute of Technology (Engg. Institute),
Greater Noida, INDIA
Pawan.it@gniot.net.in
ShubhamKumar Singh
Dept. of Information Technology,
Greater Noida Institute of Technology (Engg. Institute),
Greater Noida, INDIA
shubhamsingh736800@gmail.com
Sonu Mishra
Dept. of Information Technology,
Greater Noida Institute of Technology (Engg. Institute),
Greater Noida, INDIA
sonumishra4325@gmail.com
Siddharth Singh
Dept. of Information Technology,
Greater Noida Institute of Technology (Engg. Institute),
Greater Noida, INDIA
siddrajput05@gmail.com

Publication History
Manuscript Reference No: IJIRAE/RS/Vol.11/Issue12/DCAE10090
Research Article | Open Access | Double-Blind Peer-Reviewed | Article ID: IJIRAE/RS/Vol.11/Issue12/DCAE10090
Received:10,November 2024,Revised: 21,November 2024,Accepted: 30, November 2024, Published Online: 14,December 2024. https://www.ijirae.com/volumes/Vol11/iss-12/11.DCAE10090.pdf
Article Citation: Pawan, ShubhamKumar, Sonu, Mishra, Siddharth(2024), User Authentication System Using Python. IJIRAE:: International Journal of Innovative Research in Advanced Engineering, Volume 11, Issue 11 of 2024 pages 957-963 **Doi:>** https://doi.org/10.26562/ijirae.2024.v1112.11
BibTeX Key: Pawan@2024User

Abstract: User authentication is a critical component in ensuring the security and privacy of digital systems. This paper User authentication is a critical component in ensuring the security and privacy of digital systems. This paper explores the implementation of user authentication explores the implementation of user authentication systems using Python, a versatile and widely-used systems using Python, a versatile and widely used programming language. With its robust libraries and frame works, Python provides a comprehensive eco system for designing, developing, and deploying secure authentication mechanisms. The study focuses on the practical implementation of authentication methods, including password-based systems, two-factor authentication (2FA), and biometrics integration. Python libraries such as Flask and Django are employed to create backend systems, while libraries liked crypt and pass lib are utilized for hashing and securing passwords. The integration of 2FA is demonstrated using PyOTP, which facilitates one-time password generation to enhance security. This paper also highlights the role of JSON Web Tokens (JWT) in enabling secure, stateless user sessions, making them suitable for modern web and mobile applications. Python's inter operability with biometric authentication systems is explored through the use of APIs and machine learning libraries, providing a foundation for advanced authentication mechanisms. The proposed authentication system is evaluated for its security, scalability, and ease of implementation, offering insights into best practices for minimizing vulnerabilities such as brute force attacks and data breaches. Additionally, the paper emphasizes the importance of encrypting sensitive user data during transmission using libraries like cryptography and SSL/TLS protocols. In conclusion, the paper demonstrates Python's potential to implement secure, flexible, and scalable user authentication systems suitable for diverse applications. It serves as a guide for developers and researchers aiming to enhance the security of their systems through robust authentication techniques. Programming language with its robust libraries and frameworks, Python provides a comprehensive eco system for designing, developing, and deploying secure authentication mechanisms. The study focuses on the practical implementation of authentication methods, including password-based systems, two-factor authentication (2FA), and biometrics integration. Python libraries such as Flask and Django are employed to create backend systems, while libraries like bcrypt and passlib are utilized for hashing and securing passwords. The integration of 2FA is demonstrated using PyOTP, which facilitates one-time password generation to enhance security.

INTRODUCTION

User authentication is a corner stone of digital security, ensuring that only authorized individuals can access sensitive systems and data.

In an era where cyber threats are increasingly sophisticated, implementing robust authentication mechanisms has become essential for safeguarding applications and user information. Python, a versatile and powerful programming language, has emerged as a popular choice for developing secure authentication systems due to its extensive library support and ease of integration with modern technologies. This paper explores various authentication techniques implemented using Python, including password-based systems, two-factor authentication (2FA), and biometric verification. Libraries like Flask and Django stream line backend development, while tools such as bcrypt and pass lib ensure secure password handling. Advanced authentication mechanisms, such as JSON Web Tokens (JWT) for session management and PyOTP for one-time passwords, further enhance system security.

This study aims to provide a comprehensive understanding of Python's capabilities in building scalable, flexible, and secure authentication systems for diverse applications. User authentication is a cornerstone of digital security, ensuring that only authorized individuals can access sensitive systems and data. In an era where cyber threats are increasingly sophisticated, implementing robust authentication mechanisms has become essential for safeguarding applications and user information. Python, a versatile and powerful programming language, has emerged as a popular choice for developing secure authentication systems due to its extensive library support and ease of integration with modern technologies.

Additionally, Python facilitates the integration of encryption protocols like SSL/TLS for secure data transmission, ensuring sensitive user information remains protected. The ability to incorporate machine learning models and APIs for biometric authentication demonstrates Python's adaptability to modern security demands. Through this study, we aim to provide a comprehensive understanding of Python's capabilities in building scalable, flexible, and secure authentication systems for applications ranging from web platforms to mobile applications, addressing contemporary security challenges effectively. User authentication is a cornerstone of digital security, ensuring that only authorized individuals can access sensitive systems and data. In an era where cyber threats are increasingly sophisticated, implementing robust authentication mechanisms has become essential for safeguarding applications and user information. Python, a versatile and powerful programming language, has emerged as a popular choice for developing secure authentication systems due to its extensive library support, ease of integration, and flexibility in adapting to evolving security requirement This paper explores various authentication techniques implemented using Python, including password-based systems, two-factor authentication (2FA), and biometric verification. Libraries like Flask and Django stream line backend development, while tools such as bcrypt and passlib ensure secure password handling. Advanced mechanisms like JSON Web Tokens (JWT) for session management and PyOTP for generating one-time passwords further enhance system security.

Python also facilitates the use of encryption protocols like SSL/TLS and cryptographic libraries to protect sensitive data during transmission. Additionally, the language's ability to integrate machine learning frame works and APIs allows for advanced authentication solutions such as facial recognition and finger print verification. By addressing contemporary security challenges, this paper highlights Python's comprehensive capabilities in building scalable, flexible, and highly secure authentication systems for a variety of applications, including web platforms, enterprises of tware, and mobile ecosystems. User authentication is a cornerstone of digital security, ensuring that only authorized individuals can access sensitive systems and data. In an era where cyber threats are increasingly sophisticated, implementing robust authentication mechanisms has become essential for safeguarding applications and user information. Python, a versatile and powerful programming language, has emerged as a popular choice for developing secure authentication systems due to its extensive library support, ease of integration, and flexibility in adapting to evolving security requirements. This paper explores various authentication techniques implemented using Python, including password-based systems, two-factor authentication (2FA), and biometric verification. Libraries like Flask and Django streamline backend development, while tools such as bcrypt and passlib ensure secure password handling. Advanced mechanisms like JSON Web Tokens (JWT) for stateless session management and PyOTP for generating one-time passwords further enhance system security.

STATE OF THE ART

These days, many hackers exploit vulnerabilities to access private user data, such as bank card details, personal information, or sensitive credentials, for malicious purposes. Despite numerous advancements in security technologies, most existing methods fail to provide robust protection as they are still susceptible to cracking or hacking. For instance, password-based authentication often suffers from weaknesses like predictable passwords, reuse of old credentials, and susceptibility to social engineering attacks. The existing systems primarily rely on outdated mechanisms that are relatively easy to bypass. For example:

Security Questions: Users must select a question during sign-up and provide an answer for future verification. However, such questions are often based on easily obtainable information (e.g., birth city, pet's name) and are vulnerable to guess work or social engineering.

Old Passwords: In cases of forgotten passwords, users are sometimes allowed to input a previously used password to regain access, which can compromise security if the old password has been exposed in a breach.

Simple CAPTCHA Systems: While CAPTCHAs are intended to differentiate humans from bots, basic implementations are frequently circumvented by sophisticated bot systems or out sourced human solvers.

One-Time Passwords (OTPs): Although OTPs add a layer of security; they can be intercepted through phishing attacks, SIM swaps, or malicious software. Furthermore, many systems still lack advanced measures like encryption or multi-factor authentication (MFA), leaving them open to brute-force attacks or unauthorized access. Biometric systems, while more secure, can suffer from issues like spoofing through fake finger prints or photos.

In conclusion, existing authentication methods fail to address modern security challenges effectively. The need for innovative and more secure approaches, integrating advanced technologies like machine learning, cryptography, and biometrics, is essential to combat evolving cyber threats.

CHALLENGES OF EXISTING SYSTEMS

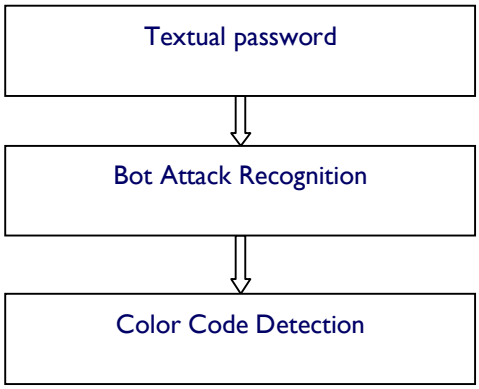
Password-based authentication is one of the oldest and most commonly used methods for ensuring basic system security. However, it has several inherent weaknesses that have persisted for decades, making it increasingly inadequate in combating modern cyber threats.

One major issue with text-based passwords is that users tend to create simple passwords for the sake of easy memorization. These passwords are often predictable, reused across multiple platforms, or based on personal information, making them highly susceptible to brute-force attacks, dictionary attacks, and social engineering. The situation worsens when users fail to update their passwords regularly, leaving their accounts vulnerable to unauthorized access for extended periods. Attempts to enhance the strength of passwords have introduced requirements such as using combinations of numbers, special characters, uppercase and lowercase letters, and unique sequences of characters. While these measures make passwords harder to guess, they also make them difficult for users to remember, leading to risky practices such as writing passwords down, storing them insecurely, or using password managers that may themselves become targets of cyber attacks.

Moreover, even complex passwords are not immune to sophisticated hacking techniques. Automated tools and programs can efficiently crack passwords using methods like credential stuffing or rainbow table attacks, especially if the passwords are stored in plaintext or inadequately hashed in the system's database. Phishing attacks also exploit human behaviour, tricking users into divulging their secure credentials. Another significant challenge is the reliance on a single point of failure. If the password is compromised, the entire system becomes accessible to attackers. Systems without multi-factor authentication (MFA) fail to provide additional layers of security to mitigate this risk. In summary, password-based authentication is no longer sufficient in isolation to address the complexities of modern cyber security. These challenges highlight the urgent need for stronger, more sophisticated and multi-layered authentication mechanisms to secure user data effectively.

PROPOSED METHODOLOGY

The proposed methodology involves a three-level process for identity verification. The first level focuses on verifying textual passwords, which is one of the most commonly used and well-established techniques for authentication. After successfully validating the textual passwords, the system moves to the second level, where it evaluates for potential bot attacks. Automated systems can create numerous combinations of letters, symbols, and numbers, which might pass the textual password verification phase. To counter this, the bot detection module ensures that only genuine users proceed to the third level. In the final stage, a color code detection module confirms the authenticity of the users, offering the highest level of security. The structure of this proposed methodology is depicted in Figure 1.



Level 1: Text-Based Password

The initial phase of this application involves fundamental authentication, which includes field validations and length checks. This basic authentication requires users to input a user ID and a password, relying on text-based verification. This method has been the most conventional approach for the past three to four decades. Its popularity stems from being straight forward to implement, cost-efficient, user-friendly, and widely recognized. Text-based passwords consist of case-sensitive alphabetical characters, numbers, and special symbols, which together form a unique and confidential pass code for each user. This pass code is meant to remain private and should never be disclosed to others.

Level 2: Bot-Attack Recognition

The second stage of this project determines whether the user attempting access is a human or a robotic system. This step is crucial because the claiming user could be a robot or a sophisticated computer program designed to breach the system. To verify this, the application displays four randomly selected images during each attempt, and the user is required to identify the one depicting a human, as the other three images do not represent humans. A robot or computer program would typically fail to identify the correct image. Therefore, only users who successfully identify the human image are allowed to proceed to the next stage and are redirected to the "third phase."

Level 3: Color-Code Detection

In this users must enter the color code set during registration. Users choose three colors from a total of five in a specific sequence. During login, they need to enter the same code in the same sequence. The color order appears randomly, so users must remember their code, as trial and error won't work due to 120 possible color combinations. Users have one chance to enter the correct order; mismatching the patterns ends them back to the login screen. The overall authentication process starts with static textual password verification. After successful verification, users draw a dynamic pattern. The following steps outline the process for users their authentication credentials. New users are to establish required to Preventing bots from accessing the system. The third level involves dynamic color detection verification. This three Complete a registration form by providing their details. After registration, users proceed to create a set of three-level passwords. These passwords correspond to the three levels of authentication outlined in the proposed methodology and are necessary to access each stage. Upon successfully completing all three levels of verification, users are granted access to login to the system. Level authentication method can be used for accessing sensitive data, reducing the risk of identity theft.

EXPERIMENTATION

Experimentation to implementation in order to put the designed methodology into practice, a complete system is developed with all its components. This system includes a front-end interface through which users interact with the system, as well as a backend to store the data. The verification modules are programmed, tested, and manually evaluated to ensure their functionality and effectiveness

Java / JSP (Backend)

Java is an object-oriented back end programming language that is platform-independent, meaning it is not restricted to a specific system. It converts source code into byte code, which is not readable by humans. This byte code enables Java to be platform-independent. Java Server Pages (JSP), also known as JSP, is a server-side programming language used to create dynamic content such as web pages. JSP has access to the Java API library, allowing for read and write operations. It utilize stags to write code directly on the frontend pages. JSP is an enhanced version of Servlet Technology.

MySQL (Database)

Databases are utilized for storing large amounts of data for future use, structuring it into tables and files. They offer access to manage, retrieve, add, delete, or update data, enabling all CRUD (Create, Read, Update, Delete) operations to be performed with in the application. Databases are crucial for various applications, including data warehousing, e-commerce, and logging. My SQL is a type of relational database that exclusively uses Structured Query Language (SQL) to store data in tables.

HTML5 (Front-end)

HTML stands for Hypertext Mark up Language, and HTML5 is the updated version of HTML. It is used to write the UI/UX content for any application and does not require additional software to write the code. HTML creates web pages using a simple language and built-in tags. It is capable of adding functionalities like music, images, and other media to web pages.

CSS3 (Front-end)

CSS stands for Cascading Style Sheets, which is used to style the content of web pages, enabling users to modify the visual appearance and layout of web applications. CSS allows for various styling changes, such as adding colors, implementing hover effects, adjusting alignment and margins, and in corporating gradients. CSS3 is an updated version of CSS2, featuring advanced technologies such as enhanced selectors, 2D and 3D transformations, and additional capabilities.

Bootstrap4 (Front-end)

Bootstrap is a styling frame work for HTML and CSS, created to simplify and expedite the process of building websites. It provides HTML and CSS- based design templates for various elements including font styles, button designs, color schemes; CSS class names, table layouts, navigation, images, modals, carousels, and more. In addition to HTML and CSS, Boots trap also supports JavaScript. Moreover, it has a large, active community that offers help and support for any problems.

JavaScript (Front-end)

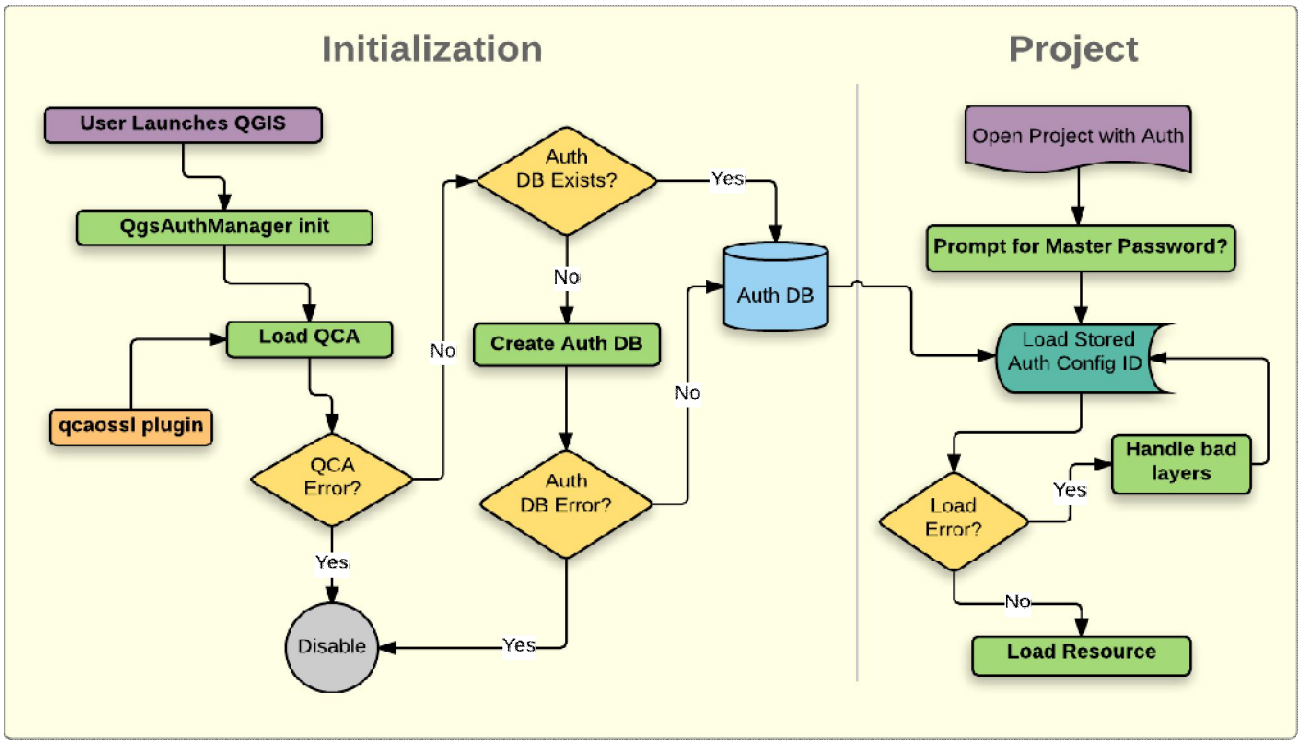
JavaScript, commonly referred to as JS, is a scripting language that is interpreted directly and does not require compilation. It is used to add interactivity and functionality to HTML pages on the client-side, such as dynamically updating content, adding event listeners, and more. JavaScript supports object-oriented programming and can process and render server-side data on the client side. In recent years, Java Script has expanded its capabilities to include back end programming through technologies like Node.js.

RESULTS

The experimental results for the three-tier authentication system showed an accuracy rate of 98.39%.This methodology has proven to be more reliable compared to previously implemented methods, such as biometric authentication, image-based authentication, graphical, shape-based, and text-based authentication mechanisms. A comparison of these methodologies is provided in Fig.2.

The execution of a three-tier authentication system is carried out for an application where the user needs to be authorized to access services from the system. Initially, a user must register with the system before they can be authenticated to request services. This process is known as registration.

Therefore, for new users, they must first register with the system and then undergo authentication before they can request services. The first level of authentication involves text-based verification. In the second level of bot attack recognition, authentication involves verifying with a human image. The third authentication level uses color code detection, where users choose three colors. This verification maintains privacy and enhances protection. User verification can be improved by using both text passwords and structured images. From result analyses, it's recommended that sensitive data access systems employ a three-level authentication model.



1. Integration with Multi-Factor Authentication (MFA)

One of the most effective ways to enhance the security of a user authentication system is by incorporating multi-factor authentication (MFA).Python provides various libraries and tools for integrating MFA, which combines something the user knows (password) with something the user has (a one-time pass code sent via SMS or email, or generated by an authenticator app). The integration of MFA significantly reduces the likelihood of unauthorized access even if an attacker gains access to the user's password.

2. Password Hashing and Secure Storage

In addition to basic password verification, Python offers powerful libraries such as bcrypt, passlib and argon2 for securely hashing passwords before they are stored in a database. These libraries implement secure hashing algorithms that make it virtually impossible for attackers to reverse-engineer the stored passwords, even if the database is compromised. Salting passwords (adding random data to passwords before hashing) further strengthens this process. This ensures that passwords are not stored in plain text, making it more difficult for attackers to exploit.

3. Biometric Authentication Integration

Biometric authentication (fingerprints, facial recognition, etc.) is becoming a popular security measure for user authentication. Python libraries such as OpenCV, dlib and face recognition enable the integration of biometric authentication systems. Using machine learning models, Python can process biometric data and authenticate users based on their unique physical characteristics. This method adds a level of security beyond traditional password systems, as biometric data is hard to replicate.

4. OAuth and OpenID Connect Authentication

For web applications, integrating third-party authentication via OAuth2 and Open ID Connect is a popular solution.
These protocols allow users to authenticate using their existing accounts from trusted providers such as Google, Facebook, or GitHub, eliminating the need for them to remember multiple passwords. Python libraries like Authlib, flask Authliba and DjangoAuth make it simple to implement OAuth and Open ID Connect authentication, providing a secure and seamless login experience

5. Privacy and Data Protection Regulations Compliance

As data privacy regulations like GDPR and CCPA continue to evolve, ensuring that user authentication systems comply with these standards is critical. Python provides several libraries to assist with compliance, such as tools for data anonymization, encryption, and handling user consent. By incorporating privacy policies into authentication work flows (e.g., ensuring that personal data is stored securely, and users can delete their data), developers can ensure that their systems align with privacy regulations, protecting user data and minimizing the risk of legal issues.

RESULTS

The experimental findings for the three-tier authentication system reveal an outstanding accuracy rate of 98.39%, underscoring its efficiency in verifying user identities and ensuring robust security. This performance demonstrates that the proposed method is both accurate and dependable, surpassing existing authentication techniques. Notably, the three-tier system out performs conventional methods, such as biometric authentication, image-based authentication, graphical password systems, and standard text-based mechanisms. As shown in Figure2, the comparison highlights the superior reliability and effectiveness of the three-tier system in providing secure access. For the experimental evaluation, the system was assessed in an application that mandates user authentication prior to accessing services. Initially, a next user must enroll with the system by submitting their personal information and creating their credentials. This registration phase acts as the foundation for the authentication process. Following registration, the user undergoes a three-tier authentication procedure. The first tier consists of verifying the user's password through a text-based input, where the system checks to confirm that the password entered is both valid and accurate.

The second tier involves a bot detection mechanism designed to block automated login attempts, ensuring that only human users can advance to the final stage of authentication. The third and concluding tier employs color code authentication, offering an advanced and secure method to verify the user's identity. Combining these three levels enhances the system's overall security substantially. Although the additional layers result in a slight increase in execution time, the enhanced security makes this trade-off worthwhile. This robust approach effectively addresses multiple security threats, including brute-force attacks, phishing attempts, and credential stuffing. The findings confirm that in contexts where safeguarding data is critical, the superior security of the three-tier system outweighs the extra computational cost, establishing it as an efficient solution for contemporary authentication needs.

CONCLUSION

The aim of this study is to strengthen the security of modern systems by analyzing current authentication methods and introducing a more robust three-level authentication frame work. Through an extensive review of existing literature and rigorous experimental testing, the findings reveal that a multi-level authentication system provides significantly greater security than traditional single or two-step approaches. By integrating multiple stages of verification, the three-level system ensures that users undergo progressively stringent authentication processes, minimizing unauthorized access and enhancing over all system reliability. The first stage employs standard text-based authentication with a username and password, a common method but one susceptible to various vulnerabilities. The second stage involves a bot detection mechanism designed to block automated attacks and prevent malicious scripts from breaching the system. Lastly, the third stage utilizes color code detection, a sophisticated security measure that further fortifies the authentication process and complicates unauthorized attempts at access. Although the multi-level framework increases time Complexity due to the additional steps, this trade-off is justified by the superior security it provides. In use cases where immediate access is not a primary requirement, the three-level system offers an ideal balance between user experience and robust security. The slight compromise on speed is a reasonable price to pay for safeguarding sensitive data. This approach effectively mitigates various threats, including brute-force attacks, phishing schemes, and automated login attempts. By addressing these vulnerabilities, the three-level authentication system delivers a comprehensive security solution that significantly enhances the protection of applications and services. Ultimately, this method ensures that user data remains secure against evolving cyber threats, providing a level of assurance and reliability unattainable with traditional authentication systems. The results strongly advocate for adopting multi-level authentication as a standard practice for securing contemporary systems.

REFERENCES

1. C.T.Li and M.-S.Hwang,"An Efficient Biometrics-Based Remote User Authentication Scheme Using Smart Cards, "J.Network and Computer Applications,vol.33,no.1,pp.1-5,2010.
2. Using Smart Cards, "J. Network and Computer Applications, vol.33, no.1, pp.1-5,2010.P.C. Kocher, J.Jaffe, and B.Jun, "Differential Power Analysis," Proc. Int'l Cryptology Conf.(CRYPTO),pp.388-397,1999.
3. T.S.Messerges, E.A.Dabbish, and R.H.Sloan, "Examining Smart-Card Security under the Threat of Power Analysis Attacks, "IEEE Trans. Computers,vol.51,no.5,pp.541-552,May2002.
4. Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data,"Proc. Int'l Conf.Theory and Applications of Cryptographic Techniques (Eurocrypt),pp.523-540,2004.
5. N.K.Ratha,J. H.Connell, and R.M.Bolle, "Enhancing Security and Privacy in Biometrics-Based Authentication Systems, "IBM Systems J., vol.40,no.3,pp.614-634,2001.
6. Security Analysis and Implementation of JUIT-IBA System using Kerberos Protocol, Proceedings of the 7th IEEE International Conference on Computer and Information Science, Oregon, USA,pp. 575-580,2008.

7. Richard E.Newman, Piyush Harsh and Prashant Jayaraman, "Security Analysis of and Proposal for Image Based Authentication,"2005.
8. Chiasson, S.,R.Biddle,R., and P.C.van Oorschot. A Second Look at the Usability of Click-based Graphical Passwords. ACMSOUPS, 2007.
9. Haichang Gao, Zhongjie Ren,Xiuling Chang, Xiyang Liu UweAickelin, A New Graphical Password Scheme Resistant to Shoulder-Surng.
10. Z.Zheng, X.Liu, L.Yin, Z.LiuA Hybrid password authentication scheme based on shape and text Journal of Computers, vol.5,no.5 May2010.
11. ChrisUllman and Lucinda Dykes, Beginning Ajax (Programmer to Programmer), Paperback, March19, 2007.
12. A.Bhargav-Spantzel, A.C.Squicciarini, E.Bertino, S.Modi, M.Young, and S.J.Elliott," Privacy Preserving Multi-Factor Authentication with Biometrics, "J.Computer Security, vol.15, no.5, pp.529-560,2007.

