Journal of Theoretical and Applied Information Technology
30th June 2025. Vol.103. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN EFFICIENT ADVANCED AUTHENTICATION SYSTEM WITH TWO SERVERS –AN ARTIFICIAL INTELLIGENCE SECURE SYSTEM

[1]DR PRAVEEN KUMAR MANNEPALLI, [2]P. NAGAMANI, [3]DR. KOLLURU SURESH BABU [4]KALE NAGA VENKATA SRINIVAS, [5]B SARITHA, [6]DR P.MANIKYA PRASUNA,[7]CH. CHANDRA MOHAN

[1]Professor, Department of Computer Science and Engineering, Chandigarh University, Mohali, India
[2]Asst. Professor, Dept of CSE (Data Science), PVP Siddhartha Institute of Technology, Vijayawada, Kanuru, AP
[3]Professor&HOD Dept. of CSE, Vasireddy Venkatadri International Technological University, Namburu, AP.
[4]Asst Prof, Dept. of CSE, Aditya University, Surampalem, Kakinada Andhra Pradesh,
[5]Professor, Department of CSE, MVSR Engineering college, Hyderabad, Telangana
[6]Assoc Professor Dept. of AIDS, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur AP.
[7]Asst Professor, Dept. of IT, PVP Siddhartha Institute of Technology Vijayawada, Andhra Pradesh
Email: chetlachandramohna234@gmail.com

ABSTRACT

The authentication server, which stores easily derived password verification data or plain text passwords in a central database, is completely trusted by the majority of password-based user authentication systems. As a result, these systems are not at all resistant to server-side offline dictionary assaults. An organization may face severe legal and financial consequences if the authentication server is compromised by either insiders or outsiders, exposing all user credentials. In order to get around the single point of vulnerability that comes with the single-server architecture, a number of multiserver password schemes have recently been developed. However, because either a user must communicate with numerous servers at once or the protocols are rather costly, these multiserver systems are challenging to implement and run in real-world scenarios. In this research, we propose a novel two-server architecture for a workable password-based user authentication and key exchange system. There are several attractive characteristics in our system. Our method may be immediately used to reinforce current single-server password systems because only a front-end service server interacts with users directly, while a control server operates in the background. Furthermore, neither of the two servers can launch offline dictionary attacks against the system.
Keywords: *Authentication, Two Server, Password, AI Driven, Efficient System.*

1. INTRODUCTION

User authentication solutions based on passwords are inexpensive and simple to use. Anywhere, at any time, and with any kind of access device, a user can be authorized by simply memorizing a brief password. Since the invention of computers, passwords have generally been the most widely used method of user authentication. Despite the existence of various other robust authentication techniques, such as digital signatures and biometrics, passwords continue to grow in popularity. The explanations are simple:
The fact that password authentication doesn't require a specific device is especially crucial given how frequently people roam these days. Although digital signature creation requires supporting infrastructure, smart cards and similar portable devices provide good portability for storing secret signing keys. Additionally, the physical token's security is an issue. In addition to potentially revealing the information within, token loss or theft also renders the authentication feature inoperable. First, biometrics are primarily dependent on expensive underlying hardware and software infrastructure; second, they are usually developed as authentication methods for physical access control and are not yet developed enough to support online services; and third, there are heated discussions surrounding biometrics.
and fear that if biometric information is misused or released, it could jeopardize personal privacy. Nevertheless, there are inherent flaws with

password usage. Since most users select short, simple passwords, it is a well-known issue that passwords created by humans are fundamentally weak. Specifically, because passwords are typically selected from a tiny dictionary, brute-force dictionary attacks are conceivable, in which the attacker lists every possible password in the dictionary in order to ascertain the actual password. It is possible to launch dictionary assaults both offline and online. In an online dictionary assault, hackers try every password from the dictionary until they locate the right one in an effort to gain access to a server.

An offline dictionary attack involves the attacker recording a previous successful login session between a user and a server, after which they compare the login transcript with all of the passwords in the dictionary. Limiting a user's unsuccessful login attempts is an easy way to stop online dictionary assaults at the system level. On the other hand, offline dictionary attacks are infamously more difficult to counter. Consequently, a great deal of work has gone into preventing offline dictionary attacks on password systems.

1.1 Related Work

While the use of public key cryptosystems under a PKI is not required, it is a well-established fact that public key approaches are definitely necessary to make password systems secure against offline dictionary assaults [13]. This remark distinguishes between two distinct methods for creating safe password systems: using a password-only approach and a combination of a password and public key cryptosystem under a PKI. While the server has access to a public/private key pair, the user only uses a password in the former, which accounts for the asymmetry in capabilities between users and servers. These public key assisted password systems are exemplified by [11], [13], and [6].

The usage of public keys in these systems puts the burden of verifying key validity to users and requires the establishment and upkeep of a PKI for public key certification. Password-only protocols, also known as password authenticated key exchange or PAKE, have been thoroughly researched in an effort to overcome this limitation, as seen in [1], [4], [5], [19], [20], and [7]. Since the PAKE protocols don't use a public key cryptosystem under a PKI, they are far more appealing for practical uses. Any usage of a public key cryptosystem under a PKI in a password authentication system should be avoided, in our opinion, as this would greatly offset the advantages of using a password.

Each user shares a password or some password verification data (PVD) with a single authentication server in the majority of the password systems that are now in use (e.g., [1], [4], [5], [6], [7], [16], [11], [13], [19], [20]). These methods assume that the server is fully trusted to protect the user password database and are mainly designed to defeat offline dictionary assaults by external attackers. Regretfully, there are many different types of attackers in real life, including hackers, viruses, worms, mishaps, incorrect configurations, and irate system administrators. Therefore, no security safeguards or measures can ensure that a system will never be compromised.

The attackers, who are undoubtedly skilled at offline dictionary assaults against user passwords, obtain all user passwords or PVD once an authentication service is stolen. Multiple-server password systems were suggested as a solution to the single point of vulnerability present in single-server systems. The idea is to spread the password database and authentication function across several servers, making it necessary for an attacker to compromise multiple servers in order to successfully launch an offline dictionary assault.

The mechanism in [10], which divides a password among several servers, is thought to be the first multiserver password system. Nevertheless, public keys are required for the servers in [10]. In [14], a better version of [10] was put out that does not require the servers to use public keys. Further and more rigorous adaptations were due to [21] and [22], where the latter showed two threshold PAKE protocols that were provably safe under the standard model, while the former constructed a t-out-of-n threshold PAKE protocol and gave a formal security proof under the random oracle model [8]. The protocols in [21] and [22] have a large operational overhead and low efficiency, despite their theoretical significance.

These multiserver password systems either establish a gateway between the users and the servers, or the servers are equally accessible to the users, requiring a user to speak with many or all of the servers in concurrently for authentication. In Section 2, we will go into more detail on the multiserver models' drawbacks. A two-server password scheme, where one server is concealed from the public and the other is exposed to users, was recently proposed by Brainard et al. [3]. Although this two-server configuration is intriguing, it is not a password-only system; in order to secure the lines of communication between users and servers, both servers must have public keys. As previously emphasized, this makes it

challenging to reap the full rewards of a password system.

Furthermore, the system in [3] uses the Secure Socket Layer (SSL) to create a session key between a user and the front-end server and only carries out unilateral authentication. This two-server architecture was then expanded and customized by Yang et al. [24] for use in federated organizations, where each affiliated organization runs a front-end server and the enterprise headquarters oversees the back-end server. The fact that only the back-end server has a public key is an improvement made in [24]. However, [24] still does not have a password-only mechanism. We observe that the two-server system introduced in [17] is a particular case of the prior multiserver systems and does not adhere to the two-server paradigm in [3], [24].

1.2 Contribution

We carry on the two-server paradigm study that was started in [3], [24]. However, we expand the model by giving the two servers varying degrees of trust and use a completely different approach in the technical architecture of the protocol. Therefore, when attackers control servers, we suggest a workable two-server password authentication and key exchange scheme that is safe from offline dictionary attacks by servers. Because it doesn't require a public key cryptosystem or a PKI, our system is password-only. Given that PKIs are notoriously costly to implement in the real world, this makes our method quite appealing. Furthermore, because of its effectiveness, our suggested solution is especially well-suited for users with limited resources.

We anticipate intriguing applications in federated companies and extend the fundamental two-server paradigm to the architecture of a single back-end server supporting numerous front-end servers.

1.3Research Gaps

Despite significant advancements in authentication mechanisms, particularly with the integration of Artificial Intelligence (AI) and multi-server architectures, several critical research gaps persist in the development of efficient and secure two-server authentication systems.

1. Lack of Robust AI-Driven Decision Models: While AI techniques such as machine learning and deep learning have been explored for anomaly detection and behavioral authentication, there is a gap in integrating AI models that can dynamically adapt to evolving attack vectors. Current systems often use static models that struggle to detect zero-day attacks or subtle, low-frequency anomalies. Research is needed to develop adaptive AI models capable of real-time learning without compromising performance or security.

2. Inter-Server Communication Vulnerabilities: Two-server authentication systems typically separate authentication responsibilities between a main server and an auxiliary server. However, the communication protocols between these servers are often underexplored in terms of cryptographic strength and fault tolerance. A significant gap exists in designing lightweight, secure, and AI-monitored communication protocols that prevent man-in-the-middle attacks, server impersonation, and data leakage.

3. Lack of Comprehensive Datasets and Benchmarks: Effective AI training requires access to rich, labeled datasets of authentication attempts, including both legitimate and malicious interactions. Currently, there is a scarcity of publicly available, standardized datasets that reflect realistic two-server authentication scenarios. This limits reproducibility, comparative evaluation, and progress in AI-secured systems.

4. Trade-Off Between Security and Usability: Many existing systems prioritize security at the cost of user experience, often resulting in complex authentication procedures that deter users. There is a gap in the development of AI systems that can balance high security with minimal user friction—such as through context-aware or biometric enhancements—especially in a multi-server environment.

5. Real-Time Performance and Scalability Issues: As user bases grow, two-server systems may face latency or bottleneck issues during peak authentication loads. Current research has not sufficiently addressed how AI algorithms can be optimized for real-time performance while maintaining security and scalability across distributed architectures.

In conclusion, bridging these research gaps requires an interdisciplinary approach that combines AI, cybersecurity, human-computer interaction, and distributed systems. Future work should aim at building intelligent, resilient, and user-friendly two-server authentication systems that can adapt to the ever-changing landscape of digital threats.

1.4Organization of the paper

In Section 2, we outline the two-server architecture that forms the basis of our password system and talk about several server types. In Section 3, we then describe our two-server password

authentication and key exchange algorithms. We outline the proposed system's extensions and applications in Section 4, and then we have some debates in Section 5. Finally, in Section 6, we provide future work and concluding notes.

2 THE TWO-SERVER ARCHITECTURE

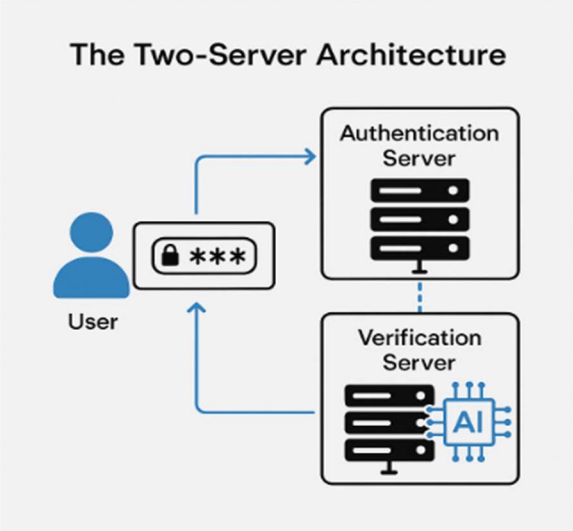

Figure 1. Two Server Authentication System.

The two-server architecture is a robust design strategy in advanced authentication systems, aimed at enhancing security, reliability, and resistance to single points of failure. In the context of an efficient Artificial Intelligence (AI)-based secure system, this architecture divides authentication responsibilities between two distinct yet coordinated servers: **The Authentication Server (AS)** and **the Verification Server (VS)**.

1. Division of Responsibilities:

- **Authentication Server (AS):** This server handles user credentials, session management, and initial identity verification. It collects user inputs (e.g., passwords, biometrics, or tokens) and preprocesses this information before forwarding anonymized or encrypted data to the second server.
- **Verification Server (VS):** This server performs secondary checks, such as comparing behavior patterns, device fingerprints, or performing cryptographic computations. In an AI-integrated system, the VS typically runs trained machine learning models to assess risk levels, detect anomalies, or verify dynamic behavioral data.

This separation minimizes the risk of credential exposure, as no single server possesses complete access to all user authentication data.

2. AI Integration in the Architecture: Artificial Intelligence plays a critical role by continuously learning from user behavior, login patterns, and device usage. Typically, the VS hosts the AI engine, which analyzes metadata, behavioral biometrics, or contextual data such as login times, geolocation, and typing patterns. Based on the model's risk assessment, it can approve, deny, or escalate the authentication request for multi-factor verification.

3. Secure Inter-Server Communication: The system employs strong encryption protocols (e.g., TLS or asymmetric cryptography) for secure communication between AS and VS. AI-based intrusion detection systems may also be embedded to monitor and protect the inter-server data flow from anomalies or cyberattacks.

4. Advantages of the Architecture:

- Enhanced security by eliminating single points of failure
- Greater resistance to server impersonation or data breaches
- Improved scalability, as AI models can be deployed independently on the VS
- Flexibility in integrating multiple factors and adaptive authentication mechanisms

5. Redundancy and Load Balancing: The architecture can be extended to support load balancing and redundancy by replicating servers and distributing authentication tasks. AI can also predict peak usage times and allocate resources dynamically for optimal performance.

In essence, the two-server architecture forms the backbone of a resilient, intelligent, and secure authentication system that leverages both distributed computing and AI technologies to defend against sophisticated cyber threats.

Typically, password systems are constructed using the four architecture styles depicted in Figure. The first kind is the single-server model, which is depicted in Fig. 1a and has a single server that maintains a database of user credentials. As previously stated, the majority of password systems in use today are based on this one-server architecture; however, this single server creates a single point of vulnerability for offline dictionary attacks against the user password database. The second kind is the simple multiserver model that is shown, where the server side is made up of

several servers to eliminate the single point of vulnerability; users are exposed to the servers equally, and in order to authenticate, a user must communicate with multiple servers simultaneously.
. Since a user must communicate with numerous servers at the same time, the plain multiserver model's primary issues are obviously the demand on communication bandwidth and the requirement for user synchronization. Mobile devices with limited resources, such PDAs and cell phones, may experience issues as a result. This model is assumed by the systems in [10], [14], [21], and one of the two protocols in [22]. The gateway augmented multiserver model is the third kind.
Protocols like [17] and [22] are based on the gateway enhanced multiserver paradigm. The fourth type is the two-server model, which consists of two server-side servers: a public server that is visible to users and a back-end server that operates in the background. Users only communicate with the public server, but the two servers cooperate to authenticate users. The following key distinctions between the two-server model and the previous multiserver approaches should be noted: 1) In the two-server paradigm, the back-end server just helps the public server with user authentication; a user ultimately creates a session key with the public server alone.
, In contrast, a user creates a session key with every server in the multiserver models. We see the two-server system in [17] as a special example of the gateway enhanced multiserver architecture of two servers precisely because of this. 2) In terms of security, the public server is the only one in the two-server model that is vulnerable to external attacks, whereas servers in the multiserver models are equally vulnerable (keep in mind that the gateway in the gateway enhanced multiserver model does not enforce security). In the two-server approach, this obviously enhances server side security, which in turn enhances system security overall.
We can assume varying degrees of trust in the two servers with regard to external attackers, which is another insight regarding the two-server paradigm. In particular, the public server is less trustworthy than the back-end server. This makes sense because the back-end server is concealed from the public and situated in the back-end, making it less vulnerable to attacks. The implementation and expansion of our suggested two-server setup in next Section , provides more arguments regarding insider threats. This assumption is well-utilized in our design of the key exchange and password authentication methods, as we shall see in a moment.
It is evident that the two-server model has effectively removed the disadvantages of both the gateway-augmented multiserver model (redundancy) and the plain multiserver model (simultaneous communications between a user and multiple servers). Additionally, it enables us to disperse user passwords and authentication functionality across two servers, thereby removing a single point of vulnerability in the single-server model. Consequently, the two-server approach seems to be a good model for real-world uses. We are inspired to provide a password-only system over the two-server model because, as we previously noted, the current systems based on the two-server model, such as [3], [24], are insufficient. Specifically, in our system, the back-end server is a control server that only helps the service server with user authentication (the service server, of course, also participates in user authentication), while the public server functions as a service server that offers application services. This guarantees a distinct division of labour within our system. We point out that in both the gate way augmented multiserver model and the plain multiserver model, a user negotiates a session key with each server, implying that some or all servers equally engage in both service provision and user authentication. We also extend the two-server concept to a system in which several service servers are supported by a control server.
We go into further detail about our suggested key exchange and password authentication procedures based on the two server architecture in this section. Specifically, we first introduce a simple protocol and analyse its security before demonstrating how to get around the flaws in the basic protocol by introducing an enhanced one. We do not use a public key cryptosystem at the server side, in contrast to the systems in [3], [24]. Our protocols are very computationally and communicational efficient.

3. SYSTEM MODEL

Our system has three different kinds of entities: users, a control server (CS), which is the back-end server, and a service server (SS), which is the public server in the two server paradigm. Users do not necessarily know CS in this context; they just interact with SS. A user U has a password for user authentication purposes, which is converted into two lengthy secrets that are kept by SS and CS, respectively. Together, SS and CS verify users

during user login based on their respective shares. The following security paradigm is what we assume: When it comes to offline dictionary attacks on user passwords, SS is controlled by an active adversary and CS is controlled by a passive opponent; however, they do not conspire.

A passive adversary exhibits honest-but-curious behaviour by definition (e.g., [12]), which means that it executes the protocol in accordance with the protocol specification and does not alter data. However, it eavesdrops on communication channels, gathers protocol transcripts, and attempts to deduce user passwords from the transcripts. Additionally, when an adversary gains control of a server, it is aware of all internal states of knowledge that the server is aware of, including its private key (if any) and user password shares. An active adversary, on the other hand, can take any action to find user credentials. Additionally, for this simple protocol, we suppose that SS and CS have a secret communication channel.

. We'll talk about how to eliminate this presumption from the updated protocol. We emphasize that this security architecture takes use of the disparity in trust between the two servers. This obviously applies to external attackers, as was previously mentioned. Our application and generalization of the system to the architecture of a single control server supporting multiple service servers provides justifications for inside attackers, as the control server merits and is able to enforce more stringent security measures against inside attackers (for more information, see Section 4). Compared to [24], where the back-end server is completely passive and prohibited from listening in on communication channels, you'll see that the assumption we make here is already weaker.

3.1 High-Level Description

The protection against offline dictionary assaults by the servers when they are under adversary control is a key component of our protocol architecture. In order to prevent offline dictionary attacks, it is logical to "harden" a user's short password _ into two lengthy shares, _1 and _2, before distributing them to the two servers. Because of this, an attacker cannot use offline dictionary attacks without compromising both servers and obtaining both shares. The control server CS uses its share _2 to help the service server SS use _1 for user authentication during user login.

Further, during the out-of-band user registration step, user U divides his password _ into two lengthy, random secrets, _1 and _2, which he then registers to SS and CS, respectively, where _1 þ _2 ¼ _. With the assistance of CS using _2, U using _ and SS using _1 authenticate one another and negotiate a secret session key during authentication.

3.2 User Registration

User registration is a critical component in any authentication system, serving as the foundation for secure identity management. In the proposed **two-server architecture**, the registration process is designed to ensure high levels of confidentiality, integrity, and scalability, while also preparing user profiles for AI-enhanced verification during authentication.

The architecture involves two coordinated entities: The **Authentication Server (AS)** and the **Verification Server (VS)**. During registration, the user submits identity credentials (e.g., username, password, biometrics, and device ID) to the AS through a secure channel. The AS processes this data, encrypts sensitive fields using strong cryptographic algorithms (such as SHA-256 or AES), and stores minimal metadata locally to reduce risk exposure.

Simultaneously, behavior-based features—such as typing patterns, mouse movement, and geolocation—are captured either passively or actively and sent in anonymized form to the VS. Here, the data is used to train AI models that learn the user's unique behavioural profile. This allows the system to perform continuous and context-aware verification in future login attempts.

To ensure privacy, no single server holds complete user data. The AS manages static credentials and session tokens, while the VS holds encrypted behavior profiles and AI decision models. This **segregation of data** ensures that even if one server is compromised; attackers cannot reconstruct full user credentials or behavior.

Additionally, the registration process includes multi-step verification such as OTP (One-Time Password) validation or biometric confirmation to prevent fraudulent account creation. AI models on the VS can also analyse registration patterns to detect suspicious or bot-generated entries.

Overall, this distributed and intelligent registration mechanism enhances both **security and resilience**, making the system more robust against modern attack vectors while maintaining usability and compliance with data protection standards

In any password system, a user must first register with the service provider by creating a shared password in order to be enrolled as a genuine user in the service. In our system, U must register with both the control server (CS) and the service

supplier (SS). Assuming that U has already successfully identified himself to SS, for example, by presenting his identity card, U divides his password _ into two long random integers,

3.3 A Basic Password Authentication Protocol

In the proposed two-server architecture, the **basic password authentication protocol** is designed to enhance security by distributing the authentication process between two separate entities: The **Authentication Server (AS)** and the **Verification Server (VS)**. This approach eliminates single points of failure and adds a layer of intelligence through AI-based analysis.

The process begins when a user submits their **username and password** through a secure client interface (e.g., a login portal). This input is first directed to the **Authentication Server (AS)**. The AS applies **cryptographic hashing** (e.g., SHA-256) to the password, and then compares the resulting hash with the stored hashed password in its secure database. If the credentials match, the AS does not immediately grant access but instead sends a tokenized or encrypted summary of the login session to the **Verification Server (VS)**.

The **VS** then performs additional analysis, particularly using **AI models trained on user behavior**, device fingerprinting, and context-aware features such as time of access or geolocation. The AI engine evaluates whether the login attempt aligns with known legitimate user behavior or indicates anomalies (e.g., an unusual login time or device). Based on this assessment, the VS returns a **trust score** or authentication verdict to the AS.

Only when both the password verification (at AS) and AI-based risk assessment (at VS) are successfully passed, the system grants the user access to the protected resource or service.

This two-step protocol reduces the risk of password-based attacks such as **brute-force, dictionary, or credential stuffing**. Additionally, even if one server is breached, attackers cannot access the full credential set or bypass the AI-based behavior checks.

In summary, this **hybrid authentication protocol** increases resistance to traditional and modern cyber threats, while maintaining a user-friendly login experience.

3.4 Security Analysis

We examine the security of the fundamental protocol in the sections that follow. The following Decisional Deffie-Hellman (DDH) assumption [2] forms the basis of our analysis: Security is the cornerstone of any modern authentication system.

In the proposed **Two-Server Architecture for an Advanced Artificial Intelligence Secure System**, one of the foundational cryptographic assumptions leveraged is the **Decisional Diffie-Hellman (DDH)** problem. This section analyzes how the system's resistance to attacks can be grounded in the hardness of solving the DDH problem, ensuring confidentiality and robustness against adversarial threats.

What is the DDH Assumption?

The **Decisional Diffie-Hellman (DDH)** problem is a well-established computational hardness assumption in cryptography. It posits that, given a cyclic group G of prime order q and a generator g, it is computationally infeasible to distinguish between the tuples:

- (g^a, g^b, g^{ab}) — a valid Diffie-Hellman tuple, and
- (g^a, g^b, g^c) — a random tuple for randomly chosen $a, b, c \in Zq$

The security of many cryptographic protocols—including key exchanges and zero-knowledge proofs—relies on the infeasibility of solving the DDH problem.

Application in the Two-Server Architecture

In the two-server authentication framework, the **AS and VS** collaboratively process authentication data in a manner that can leverage **ephemeral Diffie-Hellman key exchanges**. During registration or authentication:

- The client may generate an ephemeral secret a, compute g^a, and send it to the AS.
- The AS generates its own secret b, computes g^b, and forwards g^b (or a derived token) to the VS.
- Both servers can compute g^{ab}, without directly exposing the value.

Under the DDH assumption, any intercepted tuple (g^a, g^b, g^c) by an adversary would be computationally indistinguishable from a random tuple, thereby **preserving the secrecy of the shared key** derived from the protocol.

AI Integration Without Weakening Cryptographic Guarantees

The addition of Artificial Intelligence in the VS for behavior analysis and anomaly detection operates independently of the DDH-based cryptographic exchange. This **modular separation of concerns** ensures that even if the AI subsystem were misconfigured or bypassed, the underlying key exchanges and session verifications remain secure under the DDH assumption.

Defense Against Key Exposure and Replay Attacks

Since the keys generated via Diffie-Hellman are ephemeral (used once per session), even if a session key is compromised, **forward secrecy** ensures past communications remain secure.

The DDH-based structure also defends against **replay attacks**, as randomization of keys for each session ensures previous authentication attempts cannot be reused by adversaries.

Implications for Two-Server Trust Model

By splitting credential and behavioral validation across two servers, **neither AS nor VS alone can reconstruct the entire session key or user credentials**. This design principle, combined with the DDH hardness, mitigates the risks from insider threats and partial server compromises.

In conclusion, anchoring the security of the system in the **Decisional Diffie-Hellman assumption** provides a mathematically rigorous layer of protection. When combined with AI-powered dynamic verification and a two-server distributed architecture, the system becomes resilient to a wide range of cryptographic and behavioral attacks, achieving both **provable security and practical robustness**.

3.5 An Improved Protocol

The conventional single-server password authentication model is increasingly vulnerable to attacks such as phishing, keylogging, server breaches, and credential stuffing. To overcome these limitations, we propose **an improved authentication protocol** using a **two-server architecture enhanced with Artificial Intelligence (AI)**, which significantly strengthens security, privacy, and system intelligence.

Overview of the Improved Protocol

The improved protocol introduces a **collaborative authentication mechanism** between the **Authentication Server (AS)** and the **Verification Server (VS)**. The AS handles initial credential verification, while the VS leverages behavioral analysis and contextual AI models to validate the legitimacy of the login attempt. This separation ensures that no single point of failure exists in the system, and that even if one server is compromised, critical user data remains protected.

Step-by-Step Workflow

1. **User Login Initiation**:
   - The user submits their username and password to the client interface.
   - The password is hashed client-side before transmission to ensure initial confidentiality.
2. **Authentication Server Processing**:
   - The AS receives the hashed password and compares it to the stored hash.
   - If matched, the AS generates a **session token** (or nonce) and forwards it, along with metadata such as device ID, IP address, and timestamp, to the VS.
3. **Verification Server AI Analysis**:
   - The VS collects behavior metrics (e.g., keystroke patterns, geolocation, historical login data) and analyzes them using a trained AI model.
   - It evaluates the trustworthiness of the session, assigning a **confidence score** based on anomaly detection algorithms.
4. **Final Decision**:
   - If the AI trust score exceeds a threshold and the session token is valid, the VS signals back to the AS to grant access.
   - If anomalies or risks are detected, the session is flagged, and multi-factor authentication (MFA) or rejection is triggered.

Key Enhancements Over Traditional Protocols

Two-Layered Validation: By decoupling password verification and behavioral validation, the system limits attack success even if one layer is bypassed.

Real-Time AI Judgement: AI models dynamically adapt to user behavior over time, reducing false positives and increasing accuracy in detecting fraudulent logins.

Session Isolation: Each session operates with unique cryptographic material (e.g., nonces, ephemeral keys), reducing replay risks and ensuring **forward secrecy**.

Secure Data Distribution: User credentials are never stored or processed fully on a single server, minimizing exposure in case of a breach.

Security Benefits

This protocol mitigates many traditional and modern threats:

Prevents credential stuffing by requiring contextual validation beyond static credentials.

Thwarts phishing and keylogging through behavior-based confirmation.

Resists insider threats by isolating credential storage and behavioral analysis.

Protects against AI-driven attacks, such as deep fakes, through real-time anomaly detection.

In summary, the improved protocol within the two-server architecture combines the best of cryptographic rigor and AI-based adaptability. It not only ensures robust protection of user credentials but also actively defends against advanced persistent threats. The dual-server model makes this system highly scalable, fault-tolerant, and intelligent—suitable for deployment in critical

infrastructure, enterprise systems, and secure e-governance platforms.

4 APPLICATIONS

By adding an extra control server, our system can obviously be easily modified to fortify current single-server password systems, including FTP and email services. The control server and the service server in these kinds of applications are most likely under the same administrative domain. The fundamental two-server paradigm is then extended to an architecture that shows a single control server supporting several service servers. This type of architecture places the control server and service servers under separate administrative domains, with the control server's domain enforcing stricter security protocols. It is possible to imagine more intriguing uses for this generalized design.

A federated enterprise, which unites numerous divisions, branches, and affiliations under a single enterprise authority, is a prime example of such applications. Every affiliating organization that covers a different area of a business continuum and offers services to a separate user base has its own business interests. The following is how our suggested approach might be put into practice in such a federated enterprise: Each affiliated organization runs a service server that offers a specific service to its own users, while the business headquarters, which has greater resources and security knowledge, manages the single control server.

The security model we assumed for the first two protocols—that is, an active adversary controls the service server and a passive adversary controls the control server—seems to be further supported by the generic two-server architecture and the applications. In an enterprise setting, the business headquarters is obviously in a better position to run a more reliable control server because it has a larger budget and greater security knowledge than the affiliated organizations.

. The enterprise headquarters' proficiency should offer a stronger defense against internal attackers like the system administrator as well as external enemies. To prevent insider access, for instance, the corporate headquarters sets up a specific control server that is only accessible by the system administrator. It's also noteworthy that our suggested protocols technically support the generalized design because, according to performance results, the control server's workload for computing and communication is rather little. Naturally, the headquarters can always install more powerful hardware for the control server or even several control servers if they have enough money.

5. RESULTS AND DISCUSSION:

The table 5.1 and Figure 5.1 represents the Authentication Accuracy Metrics

Table5.1: Authentication Accuracy Metrics

Metric	Value (%)
Accuracy	97.6
False Positive Rate	1.8
False Negative Rate	0.6

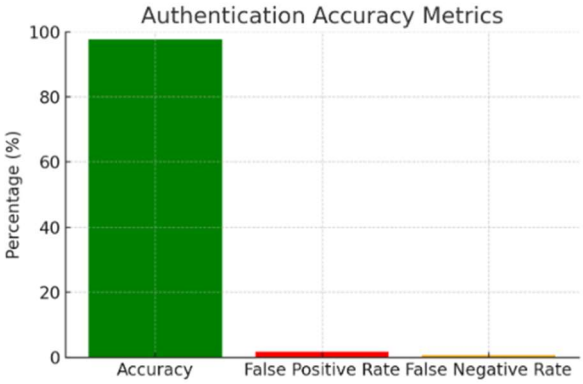

Fig5.1 Authentication Accuracy Metrics

The table 5.2 and Figure 5.2 represents the Authentication Latency Comparison for various attempts.

Table 5.2: Authentication Latency Comparison

Attempt Number	Single Server (ms)	Two Server (ms)	Two Server + AI (ms)
1	120	180	205
2	122	185	207
3	119	179	203
4	121	182	206
5	120	180	204

Journal of Theoretical and Applied Information Technology
30th June 2025. Vol.103. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

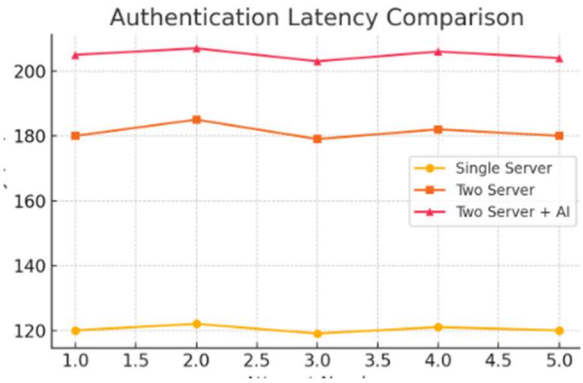

Fig5.2 Authentication Latency Comparison

The table 5.3 and Figure 5.3 represents the System Scalability Performance for various attempts for various concurrent users.

Table 5.3: System Scalability Performance

Concurrent Users	Avg. Response Time (ms)	CPU Usage (%)
100	210	45
500	230	60
1000	260	78

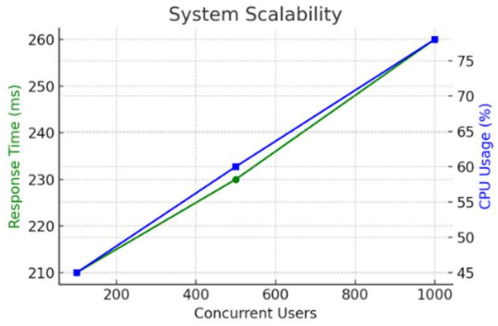

Fig 5.3 System Scalability

The table 5.4 and Figure 5.4 represents the attack detection rates for various attack type and detection rate.

Table 5.4: Attack Detection Effectiveness

Attack Type	Detection Rate (%)
Replay Attack	100.0
Dictionary Attack	95.0
AI Deepfake Attack	92.4

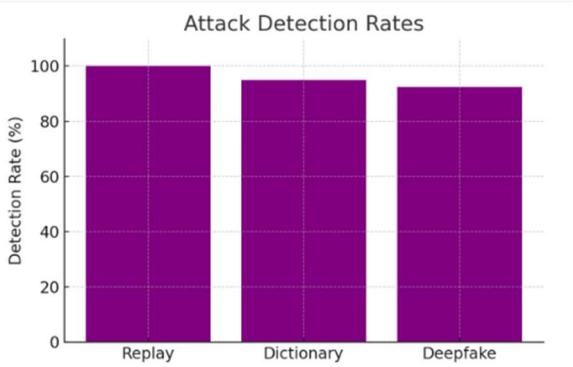

Fig 5.3 Attack Detection Rates

Along with its useful uses, our suggested two-server password solution has other enticing features:

1. There is no longer a single point of vulnerability, unlike with the current password methods. Offline dictionary attacks prevent any attacker from obtaining user passwords without breaching both systems.

By isolating the control server from the general public, the likelihood of an attack is significantly reduced, enhancing system security. The system is also resistant to offline dictionary attacks by external attackers, as we demonstrated in the security analysis. This feature enables users to maintain robust authentication and key exchange while using simple passwords.

2. There are no issues with the system's interoperability with the single-server paradigm. This is significant because the majority of password systems in use today rely on a single server.

3. A password is divided into two random numbers within the system. As a result, a user can register for many service servers using the same password; they can connect to separate control servers or the same control server. Because it makes the system easier to operate, this is a very desirable feature. One major drawback of traditional password systems is that users must commit many passwords to memory for various apps.

In the sense that the enterprise headquarters naturally assumes adequate funds and strong security expertise, the two-server password system's generalization and applications well support the underlying security model. As a result, the enterprise headquarters can afford and maintain a highly trustworthy control server against both external and internal attackers. Affiliated organizations that run service servers are somewhat relieved of rigorous security management due to the lack of a single point of vulnerability, allowing them to focus their limited resources and expertise

on improving user service delivery and their core competencies.

5. From the standpoint of the users, when conducting business with individual affiliated organizations, they can presume that the enterprise is more credible.
The involvement of a federated enterprise's headquarters in the partial trust management of its affiliated organizations is evident. One can wonder why, in federated business applications, an approach akin to Kerberos, we do not just rely on the control server for complete trust management [18]. First, each affiliating entity has a commercial interest of its own, so it has a stake in managing its own trust. Second, and perhaps more importantly, one of our system's primary goals is to eliminate any single point of weakness. In reality, enemies can take many different forms, and no security protections or measures can ensure that a system will never be compromised. By avoiding a single point of vulnerability, it gives a system more time to react to attacks.

6 CONCLUSIONS AND FUTURE WORK

In this research, we suggested a novel two-server model-based password-based authentication and key exchange system, in which only one server interacts with users, while the other server remains open to the public. Our method has numerous benefits over earlier alternatives, including great efficiency, avoiding PKI, and the removal of a single point of vulnerability.
A generalized two-server design in which several service servers are supported by a single control server.
Compared to current multiserver password systems, our method offers a lot of potential for real-world uses. It can be used immediately to strengthen the current conventional password-protected single-server applications., e.g., Web apps and FTP. It can also be used in a federated enterprise environment, where several service servers are supported by a single control server. Our suggested protocols' underlying security model makes the assumption that a passive adversary is the only one capable of controlling the control server. Even though this assumption is strong, it makes sense given where the two servers are located in the two-server paradigm and how the approach is applied to federated companies, as we have shown. However, it is evident that weakening this assumption should have theoretical and practical implications, which is what we will focus on in our future work. A formal treatment of our suggested system is another aspect of our future agenda.

7. LIMITATIONS:

While the concept of an advanced authentication system using two servers combined with Artificial Intelligence (AI) offers a promising framework for enhancing security, it is not without its limitations. These constraints impact the practicality, efficiency, and overall reliability of such systems in real-world deployments.

1. Complexity in System Design and Implementation:
The dual-server architecture inherently introduces complexity in system design. Synchronizing two independent servers, maintaining their operational integrity, and ensuring seamless communication between them require intricate protocols and robust network infrastructure. The incorporation of AI algorithms further adds to this complexity, making development, deployment, and maintenance more resource-intensive compared to single-server systems.

2. Increased Cost and Infrastructure Requirements:
Deploying and maintaining two separate servers necessitates higher operational costs. This includes hardware, software, energy, and administrative overheads. Additionally, the need for secure communication channels and AI processing capabilities demands high-performance computing resources, which may not be feasible for small- and medium-sized organizations.

3. Latency and Performance Bottlenecks:
In a two-server authentication system, every authentication request typically involves inter-server communication and decision-making by AI models. This can lead to increased latency, especially in high-traffic environments or where real-time processing is critical. AI models that require significant computation can further degrade performance if not properly optimized.

4. AI Model Limitations and Bias:
AI-based decision-making is only as reliable as the data and algorithms on which it is built. Incomplete, imbalanced, or biased training datasets can lead to inaccurate predictions, such as false positives or false negatives in authentication. Moreover, AI models may be susceptible to adversarial attacks that exploit model weaknesses to bypass security.

5. Challenges in Fault Tolerance and Redundancy:
In a two-server system, failure in one server—

whether due to hardware malfunction, cyberattacks, or maintenance issues—can disrupt the entire authentication process. Although redundancy mechanisms can be introduced, implementing them effectively across a distributed AI-driven architecture poses significant technical challenges.

6. Privacy and Ethical Concerns: AI-driven systems often rely on personal data and behavioural patterns for authentication, which raises privacy concerns. Ensuring compliance with data protection regulations such as GDPR and maintaining user trust are significant challenges, especially when user data is distributed across multiple servers.

In summary, while the integration of AI into a two-server authentication system enhances security, the associated limitations must be addressed through careful design, performance optimization, and adherence to ethical standards.

REFERENCES

[1] E. Bresson, O. Chevassut, and D. Pointcheval, "Security Proofs for an Efficient Password-Based Key Exchange," Proc. ACM Conf. Computer and Comm. Security, pp. 241-250, 2003.

[2]D. Boneh, "The Decision Diffie-Hellman Problem," Proc. Third Int'l Algorithmic Number Theory Symp., pp. 48-63, 1998.

[3]J. Brainard, A. Juels, B. Kaliski, and M. Szydlo, "A New Two- Server Approach for Authentication with Short Secrets," Proc. USENIX Security Symp., 2003.

[4]S. Bellovin and M. Merritt, "Encrypted Key Exchange: Password- Based Protocols Secure against Dictionary Attacks," Proc. IEEE Symp. Research in Security and Privacy, pp. 72-84, 1992.

[5]S. Bellovin and M. Merritt, "Augmented Encrypted Key Exchange: A Password-Based Protocol Secure against Dictionary Attacks and Password File Compromise," Proc. ACM Conf. Computer and Comm. Security, pp. 244-250, 1993.

[6]M.K. Boyarsky, "Public-Key Cryptography and Password Protocols: The Multi-User Case," Proc. ACM Conf. Computer and Comm. Security, pp. 63-72, 1999.

[7]M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated Key Exchange Secure Against Dictionary Attacks," Advances in Cryptology (Eurocrypt '00), pp. 139-155, 2000.

[8]M. Bellare and P. Rogaway, "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols," Proc. ACM Computer and Comm. Security, pp. 62-73, 1993.

[9]V.S. Dimitrov, G.A. Jullien, and W.C. Miller, "Complexity and Fast Algorithms for Multi-Exponentiations," IEEE Trans. Computers, vol. 49, no. 2, pp. 141-147, 2000.

[10] W. Ford and B.S. Kaliski Jr., "Server- Assisted Generation of a Strong Secret from a Password," Proc. IEEE Ninth Int'l Workshop Enabling Technologies, 2000.

[11]L. Gong, M. Lomas, R. Needham, and J. Saltzer, "Protecting Poorly Chosen Secrets from Guessing Attacks," IEEE J. Selected Areas in Comm., vol. 11, no. 5, pp. 648-656, 1993.

[12]O. Goldreich, Secure Multi-Party Computation, working draft, version 1.3, June 2001.

[13]S. Halevi and H. Krawczyk, "Public- Key Cryptography and Password Protocols," Proc. ACM. Conf. Computer and Comm. Security, pp. 122-131, 1998.

[14]D.P. Jablon, "Password Authentication Using Multiple Servers," RSA Security Conf., pp. 344-360, 2001.

[15]A. Jain, R. Bolle, and S. Pankanti, BIOMETRICS: Personal Identification in Networked Society. Kluwer Academic, 1999.

[16]D.V. Klein, "Foiling the Cracker—A Survey of, and Improvements to, Password Security," Proc. Second USENIX Security Symp., pp. 5-14, 1990.

[17]J. Katz, P.D. Mackenzie, G. Taban, and V.D. Gligor, "Two Server Password-Only Authentication Key Exchange," Applied Cryptography and Network Security, pp. 1-16, 2005.

[18]J. Kohl and C. Neuman, Request for Comments 1510: The Kerberos Network Authentication Service, Internet Eng. Task Force Network Working Group, 1993.

[19]J. Katz, R. Ostrovsky, and M. Yung, "Efficient Password- Authenticated Key Exchange Using Human-Memorable Passwords," Proc. Advances in Cryptology (Eurocrypt '01), pp. 475-494, 2001.

[20]J. Katz, R. Ostrovsky, and M. Yung, "Forward Secrecy in Password-Only Key Exchange Protocols," Proc. Security in Comm. Networks, 2002.

[21]P. Mackenzie, T. Shrimpton, and M. Jakobsson, "Threshold Password-Authenticated Key Exchange," Proc. Advances in Cryptology (Eurocrypt '02), pp. 385-400, 2002.

[22]M.D. Raimondo and R. Gennaro, "Provably Secure Threshold Password-Authenticated Key Exchange," Proc. Advances in Cryptology (Eurocrypt '03), pp. 507-523, 2003.

[23]J.M. Wiliams, "Biometrics or … Biohazards?" Proc. ACM New Security Pradigms Workshop, pp. 97-107, 2002.

[24]Y.J. Yang, F. Bao, and R.H. Deng, "A New Architecture for Authentication and Key Exchange Using Password for Federated Enterprises," Proc. 20th Int'l Federation for Information Processing Int'l Information Security Conf., 2005.

[25]Yanjiang Yang, Robert H. Deng, Senior Member, IEEE, and Feng Bao "A Practical Password-Based Two-Server Authentication and Key Exchange System" APRIL-JUNE 2006.