International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

A Novel Dynamic Randomized Secret Key Model Based on One-Time Password Authentication

Amanapu Yaswanth*[1], Konala Thammi Reddy[2]

Submitted: 27/04/2023 **Revised**: 24/06/2023 **Accepted**: 05/07/2023

Abstract*:* The quick improvement of the Internet encourages our lives in numerous viewpoints. Increasingly more business will be done through the Internet. Under such conditions, enough consideration must be given to data security, of which character verification is one significant issue. As per the rising advancement of smartphone devices, usage increased more and more. Like (Internet Mobile), IM banking will be progressively helpful, viable, and convenient through the new versatile correspondence frameworks. The security danger of web banking has expanded quickly as a general society regularly utilizes web-banking administrations. Among the different security methods, One Time Password (OTP) is used as a grounded strategy for authorizing security, and it is currently used in web banking administrations. With the continuous advancement of hacking innovations, it is essential to furnish clients with an ensured foundation that verifies their assets against unlawful access by authorizing control systems. There is a chance of getting OTP on the user side, so we must provide a secure channel to share the OTP from the server with its clients. In this way, the proposed model introduced with a novel concept is put forth that is more secure than the current online payment system that uses OTP. This method combines OTP with the secret key. The secret key is generated using Several different encryption techniques are used to produce the secret key at random. The Transaction password is created using a secure key and the RSA method. To prevent it from being communicated over an unsecured network and resulting in a fraudulent transaction, a copy of this password is kept on the server and is created at the user side using a mobile application. By checking the authentication code, the model finally achieves the goal of security authentication. Experiments show that OTP and Secret Key are more effective and guarantee the security feature.

Keywords: *OTP, Traditional OTP, Hash, Password.*

1. Introduction

One-time password (OTP) frameworks provide a secure verification component for logging in to a business service that uses a Secret Key that can only be used once[1]. It is generated randomly by the company or services and produced at various times. The generated secret password provides security for the freshly formed or regenerated session since it uses different numbers each time verification is attempted.

Misrepresentation Blocking: OTP verification offers a few benefits over utilizing static passwords. Because of their unique password, it decreases the danger of replay assaults.

Robot Blocking: This framework helps obstruct robots that create programmed profiles and perform tireless tests to dodge approval tests like CAPTCHA ("Completely Automated Public Turing test to distinguish Humans and Computers. ") process of generating a one-time password.

Association Information Formation: Association data refers to an irregular worth that can be aggregated from occasions, time data, etc. Contingent upon the assortment strategy, this might be a crude worth, or an arbitrary number changed over through extra cycles like encryption and hashing.

Generation algorithm: By using this cryptographic calculation is utilized to create OTP. The calculation scrambles the affiliation data to deliver a 20-byte cycle string. The algorithm is deterministic and produces an identical ciphertext from similar affiliation data.

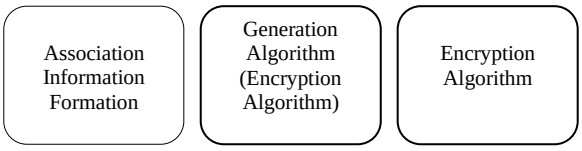

Fig 1. The OTP has been categorized into three different phases

Extraction algorithm: This algorithm removes 3-byte information of the code used for OTP[2] and is isolated into static and dynamic algorithms. This algorithm performs 'extraction data' to decide the region to isolate and includes the values from 'extraction data' and uses them in specific code text. The isolate code, 3-byte hexadecimal, had to be encoded to remember it. After eliminating the 3-byte, the code will be in [0, 16777215]; therefore, the main two numbers have some modifications or alterations. Two numbers are disposed of among these two lines, and just the lower six numbers are used (Only 0~1 for 107, just 0~6 for 106); finally, it reaches the range of [0, 999999].

Process of generation of an OTP using a cryptography algorithm. The backend server produces the Secret Key. The server generates a Secret Key to assist in creating OTP. A hash-based message confirmation code (HMAC) is created utilizing the obtained Secret Key and time, finished using the cryptographic SHA-1 algorithm. Here the server and gadget need to hold back to get granted access which is considered the boundary in the calculation. Here, the UNIX timestamp is viewed as free of the time region, i.e., time is

[1] *Department of Computer Science and Engineering. GITAM (Deemed to be University), Visakhapatnam, Andhra Pradesh, India.*
ORCID ID : 0000-0002-3121-8147
[2] *Department of Computer Science and Engineering, GITAM (Deemed to be University),Visakhapatnam, Andhra Pradesh, India.*
ORCID ID : 0000-0002-3805-9345
** Corresponding Author Email: yamanapu@gitam.edu*

determined in seconds beginning from 1st January 1970.

2. OTP Authentication Methods

2.1. Time Synchronization

The largest verification technique is time synchronization [3]. After sharing a password, a server and a customer are synced with time in this technique. When a client has to be verified and exchange the Secret Key from the server at the time data to generate value, OTP. After the customer creates and transmits an OTP incentive to the server, the server wants to create an OTP after using the same technique. It examines whether the generated two different OTPs were the same or not.

After that, those two OTP values are equal, and the client may be adequately verified. Fig 1 depicts the overall method. When a consumer and a server are already synced with time and provide a password, communication overhead can be greatly reduced. It also has the flaw of not verifying the client if the corresponding time between those two gatherings isn't correct. As a result, OTP is only valid for a limited time. The OTP value will be modified after this time.

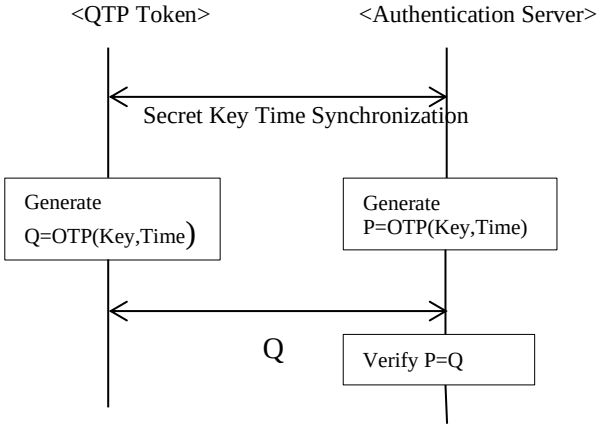

Fig 2. Time Synchronization Process Authentication

2.2. Challenge-Response to Authentication

The process of challenge-response to the Authentication test esteems the server users [4], [5], [6] to validate clients. Before confirmation, a server and a customer exchange the Secret Key. After that, the client requires verification, and the server generates test esteem and provides it to the customer. The server and the customer create the OTP esteems concurrently using a similar technique as the test esteem and the common mystery key. Fig 2 depicts the overall operation. The test reaction verification is quite safe, given that the server produces several test values each time it differs from the customers trying to validate. Be that as it may, it is helpless against correspondence assault when the test esteems are uncovered. In addition, servers and customers must communicate further more often than during the synchronization technique.

2.3. Secure/Key Authentication

A Secure/Key confirmation creates the OTP values using a hash function [7], [8]. The verification procedures are as per the following. Initially, a server and the customer communicate a mystery key. At that point, the mutual mystery key becomes an info incentive for the hash work. The server and the customer will create an incentive using this Secret Key [9], [10]. The principal hash esteem becomes an info incentive for the next hash work. The calculation rehashes this methodology for Nth times. On the customer's side, the Nth hash value is put away in the customer amassing while the server generates the last Nth hash esteem. If m is equivalent to N, the server and customer should introduce the procedure. The general procedure appears in Fig. 3.

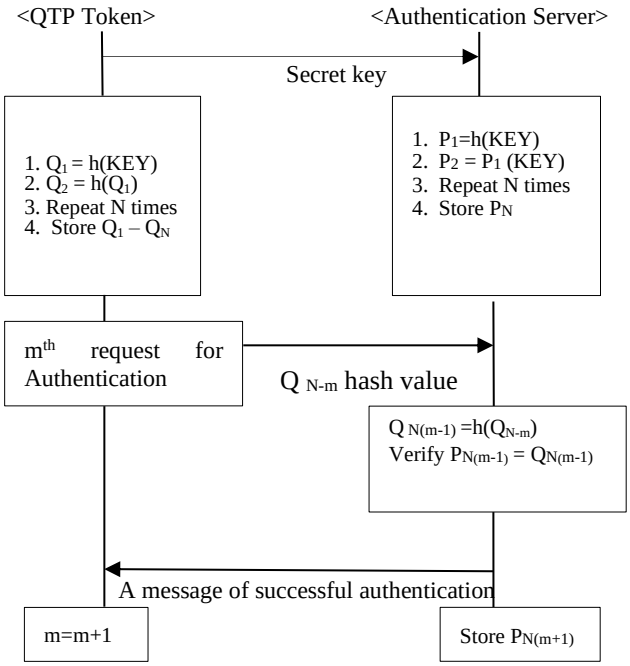

Fig 3. The process of challenge-response to the authentication

2.4. Event Synchronization

Occasionally synchronization happens in the server and customer exchange of the Secret Key, including an introduction procedure incentive [11]. Which tally esteems the quantity to its validation. When a client needs to be verified, the customer builds the checked one and produces OTP esteem with the tally to the common Secret Key. The customer sends the produced OTP incentive to the server from that point. The server expands the tallied to one and creates an OTP esteem. At that point, he verified whether the generated OTP esteem and produced OTP esteem were equivalent. The general procedure appears in Fig. 4. The tally estimations of both the server and the customer should be equivalent to fruitful verification. This strategy has an issue that if the customer creates OTP esteem but doesn't pass it to the server, the check worth would not be the same. So, the verifications will be fizzled because of various tally estimations of the server and the customer. An Outline of the scientific classification of Verification Assaults Various assaults in confirmation with counteractive measures has been proposed.

2.5. Man-in-the-Middle Assault

Common assaults come in the form of MITM. For example, a client site (bank or e-commerce) comes in between two performers and the communication goes just over the attacker. So, it mimics both gatherings and may duplicate, modify or erase some of the information traffic between

them, for example, assault on joint verification. MITM might be utilized and may not be reused to screen the information. It very well might be an aloof assault or dynamic assault. During OTP Generation from the server side, the intruder installs a third-party application in the client without knowing him. The intruder can receive the message simultaneously at the same time, leading to a lack of security.

2.6. Phishing Assaults

Phishing is a sort of assault wherein the assailant imitates the site, email, or call for a terrible reason. It is a deliberate burglary of client accreditations [8]. Phishing assaults aim to take charge of card data, email, secret words, or other delicate data. The assailant makes a site like the first one, like a financial one. DNS reserve harming empowers the client to explore the assailant's phony site.

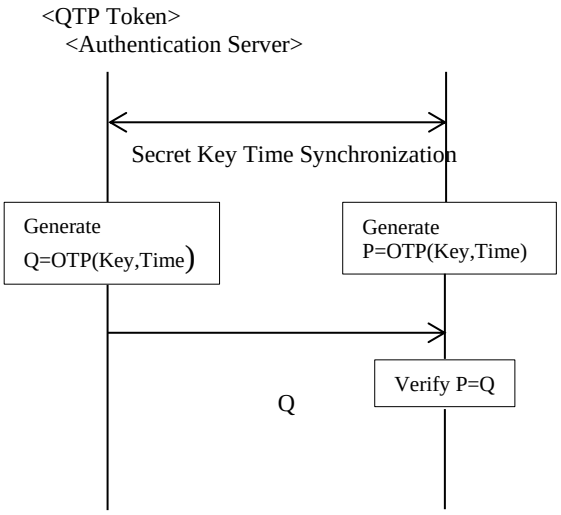

Fig 4. Process of authentication in S/key for verification

Through this assault, the intruder creates fake website links that steal complete personal computer or mobile information, including OTP messages, by clicking those links. They send those links in various ways, like messages or mail to users and sometimes, the intruder calls and collects OTP information wisely.

2.7. Insider Assault

Insider assault is a pernicious assault deliberately inside an association [12]. The association's workers offered more force and information about the climate that started such an assault. The framework directors or organization supervisors take the validation information or trade keys. OTP can easily be predictable by this assault, either in the presence or non-presence of a person. The message received from the server in some smart gadgets will be displayed in notifications and can easily be predicted.

2.8. Keylogger Assault

A key logger is computer software or code that records the client's keystrokes to obtain his secret phrase. Key loggers do not need to be programmed consistently. It is also commonly used as a piece of equipment [13]. It is commonly used for major corporate security, parental control, and other purposes; in addition to the password, OTP can be stolen easily by using this assault using different operating systems. The intruder can get the OTP whenever required by using these applications at any interval.

2.9. Malignant Code Assault

Content Youngster, basic Key lumberjack software to cutting-edge malicious Trojans might be used. Infections, Worms, Diversions, Content Youngsters, Java applet Assaults, ActiveX Controls, and others are some harmful programs. Melissa, Love Bug, Happy99, Programs Red, Sir Cam, Nimda, Prison, Back Hole, Sub7, Explore.zip, and others are examples of malicious code. This assault is difficult to differentiate [14]. The malicious code is usually concealed in emails, websites, document transfers, Java applets, and Archive content. This assault can easily retrieve OTP by installing third-party applications attached to a document, image, or suitable source, making the client believe it is an image, video, or document file. The third-party application is installed when he clicks the content, and the complete information is sealed.

2.10. Shoulder riding assaults(SRA)

Assaults utilizing social designing, for example, checking the console passage by the client or gathering his data to confirm that it was utilized for the Secret Key or structures pattern to the Secret Key. In these assaults, trust-based is mainly considered; the intruder wisely convinces the client to get the gadget and, within a short time, installs third-party applications or malicious applications and steals the information viewed from their gadget.

2.11. SQL Infusion Assault

SQL Infusion Assault is a code infusion method used to assault sites and log in with direct advantages. The ineffectively planned sites are the survivor of the assault. This assailant infused SQL orders and gained access to the data set [15]. Firewalls or IDS can't ensure information counter to SQL infusion assault. OTP can easily be retrieved by using this assault. The intruder tries to inject all database hacking applications (Kali Linux, Metasploit, etc.) into the gadget to steal OTP with or without knowing the client. In all possible manners, even the client can't see an icon in his gadget and inject SQL query for whatever he required a source of information he needed.

2.12. An Outline of the Secret Key for Secure Authentication

The security of the encryption process may be improved in several ways by using various encryption algorithms to create a Secret Key:

Multiple encryption techniques give extra protection, increasing resistance to assaults. The other algorithms can still offer security even if one algorithm is hacked or a vulnerability is found. This strategy makes it more difficult for attackers to take advantage of flaws in a particular algorithm and raises the barrier to unlocking encryption.

Protection against algorithm-specific attacks: Certain encryption algorithms could have particular flaws or vulnerabilities that attackers might exploit. By using a variety of encryption algorithms, you lessen your vulnerability to attacks created exclusively for a given algorithm.

Adaptability to shifting security environments: new algorithms are continually being created in the field of

cryptography to deal with new threats. You may develop a flexible architecture that can adapt to changes in the security environment by utilizing various encryption techniques. You may quickly switch to a more secure option if an earlier, reliable algorithm is hacked or gets out of date without compromising the encryption process as a whole.
Protection against brute-force attacks: To decode encrypted data, brute-force attacks methodically test all key combinations. Multiple encryption methods increase the computing effort needed to crack the encryption since an attacker must run brute-force assaults independently for each technique.
Compliance with security rules and standards: In some circumstances, compliance with security regulations and standards may call for several encryption methods. These guidelines frequently stress the value of using many layers of protection to secure sensitive data.

3. Proposed Method

The proposed procedure includes a pin for the cryptographic key creation stage to get OTP. The server transaction framework sends an OTP encoded and implanted at the server end with a time frame limit, and the keys are exchanged between the client and server for authentication in a secure channel. Extricating and translating are completed at the client's versatile. The secret message is epitomized and sent as a single element that forestalls potential assaults. This strategy also sticks to the five security standards proposed in.
Integrity: The OTP sends as cipher text [SMS] from the server to the client. If the substance is changed halfway, the extraction of the OTP can't be imaginable for the certifiable client, and consequently, assault is perceived. The secrecy with which these restrictions are maintained affects the degree of confidence. Pin's should be protected on the server, and the client cannot divulge their pin to anybody. If something happens, confidentiality may be jeopardized.
Authentication: The proposed strategy utilizes a pin and Secret Key for validation. The client generates the Secret Key and stores it at the server end whenever a transaction is initiated. Then the client has to validate credentials.

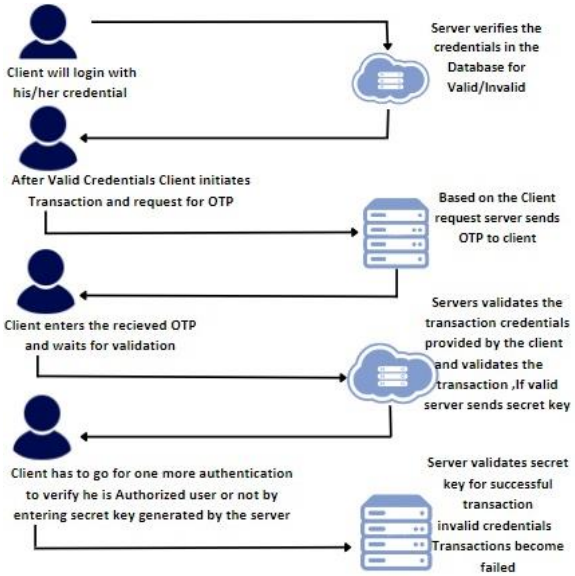

Fig 5. System Architecture

Availability: The accessibility of OTP in SMS relies upon the variables recorded underneath. Mobile device: A mobile with a mobile app is required for removing and unscrambling. The calculations are considered lightweight. Network Service Distributor: The sign strength and appropriate organization components by taking care of traffic and force of the specialist conclude at the time inside where the SMS is carried.
Limitations at the server: If the OTP approval comes up short, the proposed strategy disposes of the cycle and decreases the heap on the server. The server should have rapid handling components to deal with different solicitations produced.
Non-Repudiation: The client must enroll his credentials through e-banking administrations or offline banking mode by providing a primary device mobile number by registering with a unique id, i.e., IMSI (International Mobile Subscriber Identity). The client must enlist by visiting e-banking administrations by providing his mobile number, IMSI (International Mobile Subscriber Identity). The bank additionally secures the pin given by the client during enrollment. The client can't deny the online transaction since the pin is shipped off and should be utilized for perusing the OTP. The user must enroll one in e-banking in person with the companies by providing his cellphone number and IMSI (International Cellphone Subscriber Identity). The bank also provides the pin during enrollment. The client cannot argue against the firm's growth because the OTP SMS is delivered to this specific number, and the pin must be used to check the OTP. Fig. 5 depicts the flow of the client who had to get registered in a bank server with his credentials before he initiated an online transaction from his side. Initially, the first step is for the client to log in with valid credentials, i.e., User id and pin or password, which will be verified through servers from the banking database, allowing the process to continue. Finally, when he is ready for the transaction, the client requests the server again to send OTP. Along with the pin, the client enters a Secret Key using a set of crypto algorithms for authentication. The server validates the transaction entered by the client, valid credentials lead to a successful transaction.

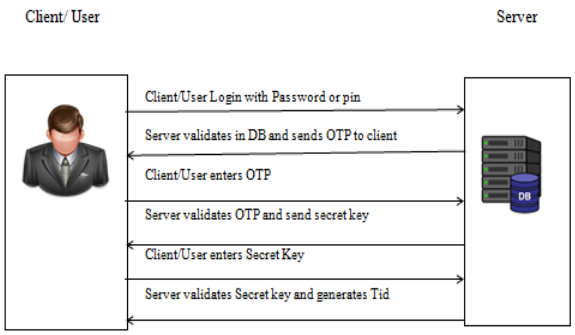

Fig 6. Process of generating OTP between Client and Server

3.1. A. Process of generating OTP

The client tries to log in with their credentials, i.e., user id and Password and verifies credentials through servers from the representative database. The server validates the credentials provided by the client. When the client is ready for an online transaction, he requests the server to generate OTP . The server will generate an OTP and send it to

validate the client. The client has to enter the received OTP and the Secret Key(pin), which is of key size N. Transaction will be completed if OTP and Secret Key (Pin) match between server and client. Transactions will fail if any one of the keys doesn't match i.e., either OTP or Secret Key or due to network failure issues during the transaction. Cryptography is used to secure OTP in the form of SMS and Secret Key for the advantages of each technique. The detailed process is explained step-by-step (Fig. 6 depicts the flow) between client and server in online transactions.

Table 1. Proportional study of vulnerability, countermeasures, authentication mechanism- pro's and con's

Vulnerability	***Strategies***	***Validation Method***	***Pro's***	***Con's***	***Standard OTP***	***Secret Key***
Eavesdropping [16]	Encrypted Password	Secured Socket Layer	Secured Process through Online	Cost affect	Yes	No
	Generated Token-based using CAS	Ruby CAS, Secure ID	Single-Sign In	It has not that much control over navigation	Yes	No
	Verification Protocols	For Internet protocol Using the Kerberos	Single-Sign on, mutual authentication	Transferring users to Kerberos is a separate database	Yes	No
Man-in-the-Middle Attack [17]	Encryption	Secured Socket Layer	Confidentiality	Performance	Yes	No
	Must build Mutual Trust	Certificate Authority – Certificates	Protection & Speed	It will be Expired & Cost-effective	Yes	Yes
	Hashing values	Hash-based Message Authentication Code	No need for a secured Socket Layer	Conflicting	Yes	No
Phishing Attacks [18]	Two-way authentication	Digital Certificates	safeguards against impostor	Vendor support, algorithm strength	Yes	No
	Avoid download from unreliable source	Digital Signatures	Authentication, Avoids imposter	Cost-effective	Yes	No
	Checking the Padlock Icon	Hypertext Transfer Protocol Secure	It Provides Confidentiality	Performing	Yes	No
Brute Force Attacks [19]	Intrusion Detection System	Honeypot	Minimalism	Hazard	Yes	No
	Test Human	CAPTCHA	Prevents of bots	Unable to get the data	Yes	No
Dictionary Attack [20]	Strong passwords	SALT Value is included in Hashed	Predicting Harder	Gentle	Yes	No
Insider Attack [21]	Access Control, Monitoring	IDS	Supervises threats inside & outside	Distinguish friend & foe	Yes	Yes
Keylogger Attack [22]	On-screen Keyboard	Microsoft OSK	Reduces the cost, space	open typing	No	No
	One time password	SafeNet one-time password	Protection	spoof able	No	No
	Ant logger	Zemana, Sandboxie	Avoid keyloggers	cost & maintenance	No	No
Malicious Code Attack [23]	Continues Updates	Automatic updates Enable	Auto management	can be compromised	Yes	No
Session Hijacking [24]	Secured Socket Layer with Cookie system	Shibboleth	access control list defined easily	Modified - resume the web server	Yes	No
	Reliable Protocol	Kerberos	Unable to enter	Replay attack	Yes	No
	Encryption	Cryptography	Privacy & safe	Consumes time	Yes	No
Shoulder surfing attacks [25]	Disturbs the attacker	Including incorrect information	safe from social Engg. Attack	Simple	Yes	No
SQL Injection Attack [15]	Modernize	Appropriate Validation	Identifies input	Time-taking	Yes	No
		Access Control strictly defined	Intruders	negotiated for claims elevated	Yes	No

3.2. A Novel Dynamic Randomized Secret Key Based on OTP Model

Input: Username, password, OTP, Secret Key
Output: Transaction id valid, Transaction id invalid.

1. Begin
2. Fn_ Login(username, password):
3. Initialize transactions on the client side
4. Fn_ generateOTP():
5. Fn_ generatesS_key (secret_key):
6. hash_object = hashlib.new(algorithm)
hash_value = hash_object.hexdigest()
hash_values[algorithm] = hash_value
return hash_values
7. number string = convertToString(number)
for each digit in numberString:
digitValue = convertToInteger(digit)
sum = sum + digitValue
return sum
8.functionaddDigitsAtPosition(secretkey, position=4):digit=convertToInteger(numberString[position])
return digit
9. Call EID=encryption (secretkey)
10. Call send_msg(mb_number,U_id)
If msg==True
11.Then DID=decryption(secretkey)
If EID==DID
Return 1//valid transaction id
else
Return 0//Invalid transaction id

3.3. Algorithm for generating Secret Key

1.import hashlib
2.def get_hash_values(input_data, algorithms):
3.hash_values = {}
for algorithm in algorithms:
4. hash_object = hashlib.new(algorithm)
5.hash_object.update(input_data.encode('utf-8'))
6. hash_value = hash_object.hexdigest()
7.hash_values[algorithm] = hash_value
return hash_values
8.NumberString = convertToString(number)
return number

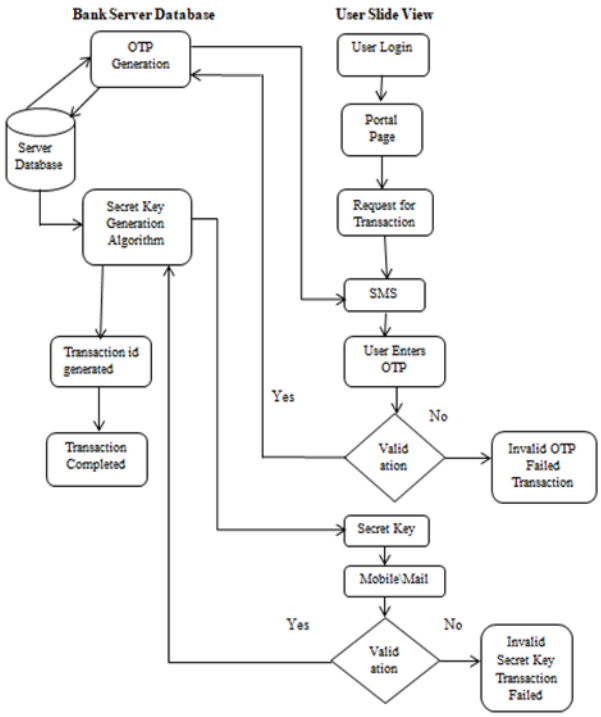

Fig 7. Flow Chart Diagram

The following steps will be processed between the client and the online portal whenever there is a need for an online transaction. The end-to-end encryption transaction process between client and server is explained in Table III. First, the user has to enter valid credentials before generating his online transaction. Once the client enters the username or id, then and after browsing his pages and if the client is ready for the transaction, from the server-side, confirmation has to be verified whether the valid credentials entered by the client or other third party person. OTP is generated from the server side, consisting of a 4- or 6-digit size randomly generated pin sent to the client; using different encryption algorithm like MD5, SHA-1, SHA-256, Argon2 etc.., a Secret Key is generated, which is sent to the client.
Conversely, the client had to enter the OTP received on his device, which is either 4 or 6-digit pin and an additional Security Key to protect from hackers or intruders. Once the OTP and Secret Key are shared with the server by the client, the server verifies the transaction id, which is decrypted; if the credentials are matched, then a successful transaction is done and keeping the status as 1, and for failed or invalid transactions, the status as 0 means any wrong entry of details.

Table 2. Data In Terms Of Validation

Username	*OTP*	*Secret Key*	*Transaction id*	*Status*
USER 1	541872.	N	263113017138	1
USER 2	682431	N	1881261730088	0

3.4. The Transaction id generated by the following steps

Step 1: To create the transaction id, the user needs to supply two inputs. A six-digit OTP created by a random

number generator, for example 541872.An illustrative secret key is 8952.

Step 2: The hacker can determine the secret key if we use it in its current form after a string of intercepted transactions. Therefore, our method uses the summation of the secret key. By adding each input from the secret key and finally the sum of input values will be considered. The total obtained in the secret key that was provided in step 1 is 24. The hacker faces a difficult challenging to decrypt the secret key as a result.

Step 3: We add the OTP and the sum of the secret key to produce the Transaction id. If we fix the position of the secret key summation as constant, hostile individuals can quickly decode the data. As a result, we do not maintain a constant for the position of secret key summation. As a result, it is always inserted in a random place. For our example, the first three position OTP value is entered, and secret key summation is chosen at 4th position and remaining values of OTP entered in a sequence. Here the OTP value is 541872 and sum of the secret key 8952 is 24. Here the summation value 24 is kept at random location we chosen 4th location in this example.

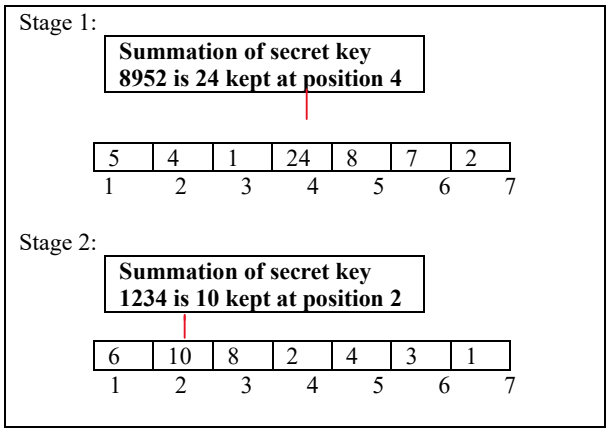

Fig 8. Summation of Secret key placed in different position's

Step 4: At this point, we wish to use the RSA technique to produce the transaction id. The size of the RSA modulus in bits, which is a constant 'bit strength' employed for this purpose, is utilized. The following lists the sub steps: Pick P and Q, two enormous prime numbers (we'll suppose P=3 and Q=11). The value of N is 33, thus calculate N=P X Q and (n) = (p-1)*(q-1). Therefore, the value of Φ (n) is 20. The decryption key, also known as the private key, is defined as d such that gcd (d, n)) =1, where gcd () stands for an integer pair's greatest common divisor. Find the encryption key, often known as the public key, which is defined as e in such a way that ed * mod ((n)) = 1, where mod is the leftover. For encryption and decryption, respectively, we have two keys: e and d. The letter P stands for plain text and represents the pass code that was created in step 3 at this point. As seen in Figure 8, it is initially encrypted using an OTP and a secret key. The plain text provided by is converted into the cypher text C

$$C = P^e \bmod n \quad (1)$$

The value of C, the Transaction id required for a successful e-commerce transaction, is obtained once encryption has been completed. This serves as an input for the subsequent decryption procedure. The cypher text C must be decoded in order to obtain the plain text P necessary to prove to the server that the user-generated transaction id they employed was obtained from the original passcode. The OTP and Secure Pin are combined to create the original Pass code number, which can be read in plain text.

$$P = C^d \bmod n \quad (2)$$

3.5. RSA Experimental Study and Example

Starting with the Passcode 54124872 computed as shown in Figure 8, which was reached by combining Steps 1, 2, and 3 above, let's begin the experimental investigation. Now, step 4 is utilized to create the transaction id, which was calculated as follows: For the sake of illustrating the result, assume that p and q are two tiny prime numbers, 3, and 11, respectively. As a consequence, N is equal to p*q=3*11=33, and (n) is equal to (p-1) *(q-1) =3*11=2*10=20. Now that we have determined the encryption key e and the decryption key d, now GCD (d, 20) = 1; let us pick d=7 to produce GCD (7, 20). We must now choose the value of e so that e*d mod (n) = 1. As a result, it is true when e = 3 because 7*e mod (20) = 1. In the following Table III shows the complete encryption and decryption process. where the RSA Algorithm is utilized to produce the transaction id from the original Pass code. The transaction id created after encryption is 263113017138 and is then used by the user for Ecommerce Transactions. The value of P is assigned 54124872.

Table 3. Obtaining the transaction id by using the RSA algorithm

Plain Text	***Encryption Module***		***Decryption Module***	
P	***P^3***	***P^3 mod 33=C***	***C^7***	***C^7 mod 33=P***
5	125	26	8031810176	5
4	64	31	27512614111	4
1	1	1	1	1
24	13824	30	21870000000	24
8	512	17	410338673	8
7	343	13	62748517	7
2	8	8	2097152	2
5	125	26	8031810176	5

4. Conclusion

During the examination, comparisons were made between various traditional OTP-based authentication mechanisms and the proposed secure key authentication techniques. Numerous potential attacks that could be applied to these OTP-based methods were identified, including Injection attacks and brute force attacks. Moreover, with the emergence of quantum computing, brute force attacks on four or six-digit OTPs have become increasingly feasible. As a result, there is a pressing need for stronger

authentication mechanisms. Thus, the study explored how OTP-based methods stack up against secure key methods, which offer an additional layer of security. Many of the attacks that can exploit OTP-based methods are rendered ineffective against secure key methods. However, to ensure enhanced protection and to mitigate the influence of quantum techniques, biometrics can be integrated in conjunction with secure keys and OTPs. This multi-layered approach further fortifies the overall security of the authentication process.

References

[1] Mulliner, C., Borgaonkar, R., Stewin, P., Seifert, JP. (2013). SMS-Based One-Time Passwords: Attacks and Defense. In: Rieck, K., Stewin, P., Seifert, JP. (eds) Detection of Intrusions and Malware, and Vulnerability Assessment. DIMVA 2013. Lecture Notes in Computer Science, vol 7967. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-39235-1_

[2] Prakash, M. V., Infant, P. A., & Shobana, S. J. (2010). Eliminating vulnerable attacks using one-time password and passtext–analytical study of blended schema. Universal Journal of Computer Science and Engineering Technology, 1(2), 133-140.

[3] S. Plaga, M. Niethammer, N. Wiedermann and A. Borisov, "Adding Channel Binding for an Out-of-Band OTP Authentication Protocol in an Industrial Use-Case," 2018 1st International Conference on Data Intelligence and Security (ICDIS), South Padre Island, TX, 2018

[4] Kushwaha, Prashant, et al. "A Brief Survey of Challenge–Response Authentication Mechanisms." ICT Analysis and Applications: Proceedings of ICT4SD 2020, Volume 2 (2021): 573-581.

[5] J. Thomas and R. H. Goudar, "Multilevel Authentication using QR code-based watermarking with mobile OTP and Hadamard transformation," 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, 2018.

[6] S. Nivetha, N. E. Elizabeth, T. P. Padmasha and I. Gohulalakshmi, "Secure authentication process in smart cards," 2016 10th International Conference on Intelligent Systems and Control (ISCO), Coimbatore, 2016.

[7] T. Kansuwan and T. Chomsiri, "Authentication Model using the Bundled CAPTCHA OTP Instead of Traditional Password," 2019 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer, and Telecommunications Engineering (ECTI DAMT-NCON), Nan, Thailand, 2019, pp. 5-8.

[8] E. Erdem and M. T. Sandıkkaya, "OTPaaS—One Time Password as a Service," in IEEE Transactions on Information Forensics and Security, vol. 14, no. 3, pp. 743-756, March 2019.

[9] ShanmugaPriya, S., A. Valarmathi, and D. Yuvaraj. "The personal authentication service and security enhancement for an optimal strong password." Concurrency and Computation: Practice and Experience 31.14 (2019): e5009.

[10] B. Maciej, E. F. Imed and M. Kurkowski, "Multifactor Authentication Protocol in a Mobile Environment," in IEEE Access, vol. 7, pp. 157185-157199, 2019.

[11] Kim, Hyunki, et al. "Analysis of vulnerabilities that can occur when generating the one-time password." Applied Sciences 10.8 (2020): 2961.

[12] Sri Sumanth, K., et al. "Pragmatic Reform to Ameliorate Insider Data Theft Detection." Confidential Computing: Hardware-Based Memory Protection. Singapore: Springer Nature Singapore, 2022. 137-148.

[13] B. Bekmezci, Ç. Eriş and P. S. Bölük, "A multi-layered approach to securing enterprise applications by using TLS, two-factor authentication and single sign-on," 2018 26th Signal Processing and Communications Applications Conference (SIU), Izmir, 2018.

[14] Bairwa, Amit Kumar, and Sandeep Joshi. "Mutual authentication of nodes using session token with fingerprint and MAC address validation." Egyptian Informatics Journal 22.4 (2021): 479-491.

[15] Abikoye, Oluwakemi Christiana, et al. "A novel technique to prevent SQL injection and cross-site scripting attacks using Knuth-Morris-Pratt string match algorithm." EURASIP Journal on Information Security 2020.1 (2020): 1-14.

[16] APRIANSYAH, YOGI. Implementing One Time Password (OTP) for Login Security on Web-Based Systems. Diss. The University of Technology Yogyakarta, 2022.

[17] Cho, Tae-Ho, and Garam-Moe Jeon. "A method for detecting man-in-the-middle attacks using time synchronization one-time password in interlock protocol based internet of things." Journal of Applied and Physical Sciences 2.2 (2016): 37-41.

[18] Bojjagani, Sriramulu, DR Denslin Brabin, and PV Venkateswara Rao. "PhishPreventer: a secure authentication protocol for preventing phishing attacks in a mobile environment with formal verification." Procedia Computer Science 171 (2020): 1110-1119

[19] Ayankoya, Folasade, and Blaise Ohwo. "Brute-force attack prevention in cloud computing using one-time password and cryptographic hash function." International Journal of Computer Science and Information Security (IJCSIS) 17.2 (2019): 7-19.

[20] Taufiq, Muhammad, and Dion Ogi. "Implementing One-Time Password Mutual Authentication Scheme on Sharing Renewed Finite Random Sub-Passwords Using Raspberry Pi as a Room Access Control to Prevent Replay Attack." 2018 International Conference on Electrical Engineering and Informatics (ICELTICs). IEEE, 2018.

[21] Azrour, Mourade, et al. "Internet of Things security: challenges and key issues." Security and Communication Networks 2021 (2021): 1-11.

[22] Papaioannou, Maria, et al. "A survey on quantitative risk estimation approaches for secure and usable user authentication on smartphones." Sensors 23.6 (2023): 2979.

[23] Mahdad, Ahmed Tanvir, Mohammed Jubur, and Nitesh Saxena. "Analyzing the Security of OTP 2FA in the Face of Malicious Terminals." Information and Communications Security: 23rd International Conference, ICICS 2021, Chongqing, China, November 19-21, 2021, Proceedings, Part I 23. Springer International Publishing, 2021.

[24] Manjula, B. Vishnuvardhanand B., and R. Lakshman Naik. "Pre-Authorization And Post-Authorization Techniques For Detecting And Preventing The Session Hijacking." International Journal of Future Generation Communication and Networking 14.1 (2021): 359-371.

[25] Binitie, Amaka Patience, Nneamaka Christiana Anujeonye, and Peace Oguguo Ezzeh. "Security against Shoulder Surfing Attack Adaptable to Feature Phones using USSD Technology.

[26] Janani, S., Dilip, R., Talukdar, S. B., Talukdar, V. B., Mishra, K. N., & Dhabliya, D. (2023). IoT and machine learning in smart city healthcare systems. Handbook of research on data-driven mathematical modeling in smart cities (pp. 262-279) doi:10.4018/978-1-6684-6408-3.ch014 Retrieved from www.scopus.com

[27] Rambabu, B. ., Vikranth, B. ., Anupkanth, S. ., Samya, B. ., & Satyanarayana, N. . (2023). Spread Spectrum based QoS aware Energy Efficient Clustering Algorithm for Wireless Sensor Networks . International Journal on Recent and Innovation Trends in Computing and Communication, 11(1), 154–160. https://doi.org/10.17762/ijritcc.v11i1.6085