A Password Manager for Post-Quantum Era

Olemilekan Rasaq Aremu[1], Samet Tonyali[2]*, Abdulkadir Köse[1]

qlh d C h jeranoepu G uoane P G U
C i d ja jeranoepu C i d ja P G U

Abstract

Quantum computing poses a threat to our classical cryptographic algorithms especially, key exchange and digital signature algorithms. The rise of this threat has also brought about the rise of solutions and the birth of a new standard called Post-Quantum Cryptography. These algorithms will be resistant against quantum attacks and will be the new standard going forward following the results from the National Institute of Standards and Technology. The goal of this paper is to investigate the progress of post-quantum cryptographic algorithms and how they perform in a pseudo-production environment through the development of a desktop application dealing with highly sensitive data. Specifically, an analysis on the performance of the key exchange and digital signature verification algorithms was conducted, and insights into the viability of these algorithms were given. The findings in this paper would aid the easy adoption of post-quantum algorithms into production environments once they become the new standard.

Keywords: *kopmq jpqi nul pkc n l du Gau t d jc a hc knepdi Gau j l oqh pek j I a d jeoi ec ep h ec j pqna kopmq jpqi ec ep h ec j pqna*

Kuantum Sonrası Dönem İçin Bir Parola Yöneticisi

Özet

Kuantum hesaplama, anahtar değişim ve dijital imza algoritmaları başta olmak üzere geleneksel kriptografik algoritmaların güvenliği için tehdit oluşturmaktadır. Shor algoritmasının yeterince ölçeklenmiş ve verimli kuantum işlemcilerde gerçeklenmesi sonucu bu algoritmaların dayandığı zor matematiksel problemler kolayca çözülebilecektir. Bu tehdidin ortaya çıkması kuantum-dayanıklı kriptografinin ilgi odağı haline gelmesine sebep olmuştur. Bu algoritmalar ABD Ulusal Standartlar ve Teknoloji Enstitüsü'nün çağrısı ile standartlaşma sürecine girmiş bulunmaktadır. Bu çalışma kuantum-sonrası kriptografi algoritmalarının ilerleyişini ve gerçek olmayan bir üretim ortamında yüksek hassasiyetteki verilerle çalışan bir masaüstü uygulaması aracılığıyla başarımının incelenmesini amaçlamaktadır. Özellikle, anahtar değişim ve dijital imza doğrulama algoritmalarının başarım analizi gerçekleştirilmiş ve bu algoritmaların uygulanabilirliğine dair bulgularımız sunulmuştur. Bu çalışmadaki bulgular, kuantum-sonrası kriptografi algoritmaları standartlaştırıldıktan sonra üretim ortamlarında benimsenmesine yardımcı olacaktır.

Anahtar Kelimeler: *Gq jpqi okjn o Gnel pkc n be j dp n a e ei hc knepi h n j dp n G l o hhai a I ag jevi o efep h i v Gq jpqi okjn o efep h i v*

1 Introduction

With the innovation of more secure methods of transmitting sensitive data, comes more ways of cracking these secure ways. Reminiscing back to World War 2 when the Enigma machine was used for secure communication, computers were subsequently developed which were able to crack this machine.

Today however, we have a similar situation with quantum computers and Shor's algorithm [1]. Modern day cryptographic algorithms are the basis of secure communications and transactions on the

* @ □ □ □□

Internet. These algorithms rely on two main problems; the integer factorization problem and the discrete logarithmic problem. As a result of these problems being very difficult to be solved, secure communication can exist, and they can be very difficult to break. However, Peter Shor, a mathematician detailed how these problems can be solved in 1994. This is known as Shor's algorithm. His algorithm was not designed to work on classical computers and instead needed quantum computers [1].

This paper would look to investigate the performance of these algorithms in a pseudo-production environment and delve into the time it takes to encapsulate and decapsulate the shared secrets, create a digital signature and comment on the ease of development.

In this section, we present the technical approaches we used in this work.

In order to understand the performance of these post-quantum cryptographic (PQC) algorithms in the context of a production environment, a password manager was developed. This application was developed as a desktop application with the following features.

Encryption and Decryption responsibility falls to the client which allows the server to be ignorant of these processes. As such, if the server is ever compromised, the encryption key is incapable of being generated.

Encryption and Decryption keys are generated on the client and not stored locally. Whenever the key is needed, the key is generated once again and used to either encrypt or decrypt the payload. This prevents attacks that can snoop through memory segments.

This application supports multiple users on a single client through JSON Web Tokens. It uses JSON Web Tokens to perform authentication and authorization mechanisms.

The application would also use post-quantum cryptography to ensure secure communication. It will use key encapsulation and digital signature algorithms. This is one of the first applications to implement these algorithms in a production or pseudo-production environment.

The technological stack chosen to develop this application was carefully thought out. It involves technology which provided ease of development without reinventing the wheel, provided a performant application and software which interacts well on all layers. The technology chosen for this project were as follows.

This is the language chosen for development on the frontend and the backend. This ensures that the same language can be written on both sides and as a superset of JavaScript, TypeScript provides type-safety which ensures that errors are caught in compile-time rather than during run-time.

Electron is a framework which allows desktop applications to be built using JavaScript, HTML and CSS. It works by embedding chromium into a desktop application. This ensures that development can be fast at the expense of a larger bundle size during packaging. Electron was used to build desktop applications for Discord, Twitch and Facebook Messenger. It offers a great developer experience.

The frontend of the application was written in React to ensure reactive user-interfaces. React is the leading frontend library and was developed by Meta.

The NodeJS JavaScript runtime powers the backend of the application. It allows for JavaScript to be written on the server and ensure the smooth interoperability of the frontend and backend by using the same language.

Express is a minimalistic framework for creating REST APIs. It provides an intuitive way to build fully equipped API endpoints and as such, the server and the proxy was written using this framework.

MongoDB was the database of choice. The decision was made due to the fact that it was a document database, easily configurable and features free AWS hosting.

The application was styled using TailwindCSS. A utility first CSS framework which provided a great developer experience while ensuring beautiful user-interfaces can be built.

ChakraUI powered the modals of the application. It provided a clean interface for creating modals and ensuring they looked the best while being performant.

Linux was the development environment of choice as it allowed the use of Liboqs library [2] which enabled the access to the post-quantum algorithms at a higher level through wrappers.

An important aspect of the application is secure communication which is provided through PQC algorithms. Before passwords are retrieved and transmitted to the client, a secure key must be exchanged between the client and server and used for this communication. This is enabled in the application through Kyber-512 [3] key encapsulation and Dilithium-2 [4] digital signature algorithms.

On login from user, the client [5] sends a request to the proxy server [7] which then engages in the key exchange with the server [6] and replies to the client with the shared secret which is used for subsequent communications. In more detail, during the key exchange, the proxy server generates a public key using the key exchange algorithm and signs it using the digital signature algorithm then sends it to the server, the server receives it and then verifies the signature and if valid, encapsulates its secret using the received public key and subsequently generates a shared secret and cipher text. The cipher text is returned to the proxy and is used to decapsulate its secret and generate the same shared secret as the server. This shared secret is then sent back to the client and stored in a database on the server side. The entire data flow is illustrated in Fig.1.

PQC algorithms not only provide quantum-safety, the NIST standards also provide additional security over classical cryptographic algorithms. A comparison between minimum key length and complexity of a successful brute-force attack shows that for key lengths ranging from 80 up until 256 bits, the NIST standards are much more resistant to brute force attacks than the integer factorization problem and discrete logarithm problem algorithms [8].

As shown in Table 1, with increasing key lengths, strength against brute force attacks grow.

Table 1. Comparison of various algorithms: Key lengths and resistance against brute-force attacks [2].

DLP	160	224	256	384	512
IFP	851	1853	2538	6707	13547
NIST	1024	2048	3072	7680	15360
C-SSI	320	448	512	768	1024
Q-SSI	480	672	768	1152	1536
C-CODE	1438	2013	2301	3451	4602
Q-CODE	2876	4026	4602	6902	9203
C-RLWE	673	921	1058	1541	2045
Q-RLWE	746	1016	1152	1688	2234

The proposed solution consists of some novelties to distinguish itself from others out there. Some of these novelties include the fact that this application is one of the first to use post-quantum algorithms in a pseudo-production environment. As these algorithms are still theoretical, applications have not begun to adapt these algorithms.

Another novelty is how the encryption and decryption mechanism are performed. The server is completely kept out of the loop and these mechanisms are only possible on the client side due to the server being unable to generate the key. This approach ensures that should the server or database be compromised, the passwords are still secure.

3 Results

In this section, we present our findings as a result of our experiments.

Following the development of the application and subsequent implementation of the PQC key exchange and digital signature verification algorithms, readings were taken to understand the performance of the process as well as other important statistics. The algorithms used are Kyber-512 and Dilithium-2. The results were taken on a computer with an Intel i5-7300hq processor running Ubuntu 20.04 on a virtual machine. The results are given in Table 2.

According to the results, key encapsulation and decapsulation are in acceptable limits. The digital signature generation and verification while taking more time are also acceptable. However, compared to the encapsulated key size, the digital signature is much larger, and that might be something to pay attention to. According to these results, these algorithms are completely viable in a production environment and once the standards have been defined, there would be no cause for alarm in terms of performance of applications.

Table 2. Performance results of PQC algorithms.

Key Encapsulation Time	92 ms
Key Decapsulation Time	0.19 ms
Encapsulated Key Size	32 bytes
Digital Signature Generation Time	22 ms
Digital Signature Verification Time	190 ms
Digital Signature Size	2420 bytes

Developing an application which needs to transfer sensitive data from server to client and vice versa is unavoidable as many applications collect and are responsible for data provided by the user, much of which are very sensitive. This approach provides a guide to these applications and how they can ensure secure communication to avoid security threats and vulnerabilities.

These advantages include;

- This means that using PQC key exchange algorithms protects against attacks from quantum computers. Although these algorithms are still theoretical and a standard has not been decided upon by the relevant authorities, once they are, it will be unwise to not implement them in all applications.
- The implementation of user authentication ensures that multiple users can coexist on a single client and perform separate key exchanges. User authentication is powered by JSON Web Tokens and transmission is performed with a valid post-quantum key and a valid access token.
- End-to-End secure applications are becoming more popular. This approach was also thought about in the design phase. The server being unaware of the payload means the passwords are secure in case the server or database is compromised.

oken.

As with all things, there are some dow9d[?(1l32( )24(n)4(t)-2 encountered during this work was the complexity of key flows. As a complex application, a multitude of keys were being exchanged, generated and stored. The complexity steadily grew as more keys were introduced. This complexity was mitigated by detailing the flow and developing a state diagram to visualize the flow before the implementation of any code.

4 Conclusion

This paper investigated the quantum computing threat and looked into how possible post-quantum standards can perform in a production environment. It was discovered that these algorithms are performant and viable and should not be an issue when implemented in a production application. Although the quantum computing threat is not a wide spread issue at the moment, now is the time to start preparing for the period it does become one. The National Institute of Standards and Technology are in the process of developing new cryptographic standards for a quantum-safe future. This process is now in the third round and hopefully once the standards are finalized, the gradual migration to these algorithms can take place.

[1] Mone Gregory. "The quantum threat." , 63.7, 12-14, 2020.

[2] D. Stebila, M. Mosca., "Post-quantum key exchange for the Internet and the Open Quantum Safe project," In Roberto Avanzi, Howard Heys, editors, Selected Areas in Cryptography (SAC) 2016, LNCS, vol. 10532, pp. 1–24. Springer, October 2017. https://openquantumsafe.org.

[3] Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G. and Stehlé, D., 2018, April. CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM. In 2018 IEEE European Symposium on Security and Privacy (EuroS\&P) (pp. 353-367). IEEE.

[4] Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G. and Stehlé, D., 2018. Crystals-dilithium: A lattice-based digital signature scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems, pp.238-268.

[5] Olamilekan Aremu, Post Quantum Password Manager, Last accessed on 31 August 2022, https://github.com/Areezy/postquantum-password-manager.

[6] Olamilekan Aremu, Post Quantum Password Manager Server, Last accessed on 31 August 2022, https://github.com/Areezy/postquantum-password-manager-server.

[7] Olamilekan Aremu, Key Exchange Proxy Server, Last accessed on 31 August 2022, https://github.com/Areezy/keyexchange-proxy.

[8] F. Borges, P. R. Reis, and D. Pereira, "A comparison of security and its performance for key agreements in Post-Quantum Cryptography," IEEE Access, vol. 8, pp. 142413–142422, 2020.