https://doi.org/10.7494/csci.2024.25.3.6082

Aldin Kovačević
Muzafer Saračević
Amor Hasić

BIOMETRICS-BASED GENERATION OF DIFFIE-HELLMAN KEY EXCHANGE PARAMETERS

Abstract *When two parties need to securely communicate over an insecure channel, Diffie-Hellman is often employed as the key exchange algorithm. This paper presents two novel approaches to generating Diffie-Hellman parameters for key exchange based on user biometrics, namely their fingerprint data. Fingerprint templates are extracted as bit strings via a fingerprint scanner and later used as inputs. In one approach, the whole fingerprint template is utilized as a user's private key. In the second approach, fingerprint data is scrambled into smaller chunks and rearranged into two strings that serve as the user's private key and the basis for prime p. Both approaches were implemented and tested experimentally. After analysis, the second approach that uses scrambled fingerprint data shows better execution times and improved security and usability considerations.*

Keywords biometrics, Diffie-Hellman, key exchange algorithm, fingerprint template, prime generation

Citation Computer Science 25(3) 2024: 1–14

1. Introduction

Traditional cryptography is used either to securely store data in a system or to transmit data over network channels. In this technique, a cryptographic key plays the most important role in protecting information. As the key is very large (e.g., 128, 192, and 256 bits for AES) and random, there is a need to securely generate and store the key in a secret place. For any communication, the secret key is either exchanged between two users (also referred to as parties) through a key-transport protocol (using public key cryptography) or established in such a way that both communicating parties have equal influence on the key that is agreed upon. Oftentimes, the involved parties need to communicate through an insecure or public channel. Hence, the need arises for an algorithm that can efficiently generate a shared secret key between two parties, without them having to reveal any sensitive data. A very popular key-exchange algorithm is the Diffie-Hellman key exchange. Published in 1976 by Whitfield Diffie and Martin Hellman, it was one of the first practical examples of public key cryptography [10].

In Diffie-Hellman, the initial parameters chosen by both parties are raised to a selected power to produce decryption keys. The components of the keys are never directly transmitted, making the task of a would-be code breaker mathematically challenging. Moreover, the two parties do not need to have prior knowledge of each other, yet they can still work to produce the secret key together. To implement Diffie-Hellman, two end users need to mutually agree on positive whole numbers p and g, and respectively choose positive whole-number personal keys a and b (both being less than p). Afterward, modular exponentiation is performed to calculate public keys A and B that the parties use to individually calculate a shared secret key x. In this algorithm, the proper choice of parameters is important for the overall security of the resulting secret key. Namely, the value of p is recommended to be at least 2048 bits due to the latest security considerations [1]. Moreover, choosing too small values for private keys a and b can also lead to the shared secret being easily deducible. Therefore, research into the optimal values for these parameters is an ongoing topic.

In this paper, a potential approach to generating Diffie-Hellman parameters is presented through biometrics – namely, fingerprint data. Biometric data is reliable for authentication and can be integrated with traditional cryptography to make it stronger [11]. Biometric data can be used to manage a cryptographic key by making a strong link between a key and the user's physiological characteristics. A cryptographic key may be derived from biometric data using any standard hash function or user-defined algorithms. While there has been research about the practicality of using biometric technology in cryptography, there has not been much focus on the generation of DH parameters using biometrics. The main aims of this paper are therefore: 1) find a practical and effective approach to generate DH key exchange parameters from user fingerprint data; 2) discuss the efficiency, practicality and limitations of biometrics-based DH parameter generation.

The rest of the paper is structured as follows: Chapter 2 presents the necessary background information regarding the Diffie-Hellman protocol and fingerprint analysis. Chapter 3 presents the relevant literature review on the topic of biometrics in cryptography. Chapter 4 describes two proposed approaches to using fingerprint data in DF parameter generation. In Chapter 5, the two approaches are analyzed and compared, with additional comments on their applicability and limitations. Lastly, the final chapter summarizes the main points of the paper and proposes future work.

2. Background information

Diffie-Hellman key exchange is a method of digital encryption that securely exchanges cryptographic keys between two parties over a public channel without their conversation being transmitted over the Internet. The two parties use symmetric cryptography to encrypt and decrypt their messages. To implement Diffie-Hellman, two end users, e.g. Alice and Bob, mutually agree on positive whole numbers p and g, such that p is a prime number and g is a generator of p. The generator g is a number that, when raised to positive whole-number powers less than p, never produces the same result for any two such whole numbers. The value of p may be large, but the value of g is usually small.

Once Alice and Bob have agreed on p and g in private, they choose positive whole-number personal keys a and b, which are less than the prime number modulus p. Neither user divulges their key to anyone. Next, Alice and Bob compute public keys A and B based on their keys according to Equations (1) and (2):

$$A = g^a mod\ p \tag{1}$$

$$B = g^b mod\ p \tag{2}$$

The two users can share their public keys A and B over an insecure channel. From these public keys, a number x can be generated by either user based on their keys. Alice and Bob can compute x using the following equations, respectively:

$$x = B^a\ mod\ p \tag{3}$$

$$x = A^b\ mod\ p \tag{4}$$

The value of x turns out to be the same according to either of the above two formulas. However, the personal keys a and b, which are critical in the calculation of x, have not been transmitted over a public medium. Because it is a large and random number, a potential hacker has almost no chance of correctly guessing x. The two users can now choose whichever symmetric encryption algorithm, and encrypt their communications using the shared key x. A biometric is defined as a unique, measurable, biological characteristic or trait for automatically recognizing or verifying the identity of an individual [16]. Today, the most common use case for biometrics is the analysis of human characteristics for security purposes. Biometric methodology for authentication is appealing because of its handiness and possibility to offer security with non-denial. The most common biometrics used in security are fingerprint, hand, iris, face and voice. Research has shown that fingerprints are suitable as long-term markers of human identity. They are detailed, unique, difficult to alter, and durable over the life of an individual, making them ideal for authentication purposes. The analysis of fingerprints for matching purposes generally requires the comparison of several features of the print pattern. These include patterns, which are aggregate characteristics of ridges, and minutiae points, which are unique features found within the patterns. The three basic patterns of fingerprint ridges that constitute the majority of all fingerprints are the loop, whorl and arch, shown in Figure 1. Other common fingerprint patterns include the tented arch, the plain arch, and the central pocket loop.

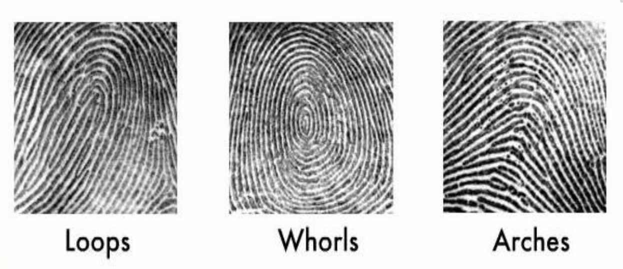

Figure 1. Most common fingerprint patterns [16]

Minutiae points are the major features of a fingerprint image and are used in the matching of fingerprints. These minutiae points are used to determine the uniqueness of a fingerprint image. A good quality fingerprint image may contain around 25 to 80 minutiae depending on the fingerprint scanner resolution and the placement of the finger on the sensor [17]. The major minutiae features of fingerprint ridges are ridge ending, bifurcation, and short ridge (or dot) as shown in Figure 2.

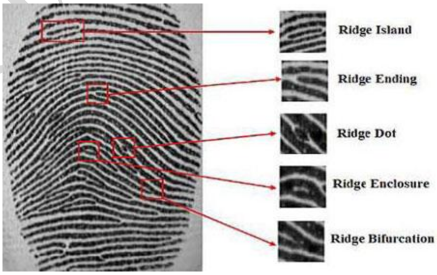

Figure 2. Most common minutiae points [16]

The ridge ending is the point at which a ridge terminates. Bifurcations are points at which a single ridge splits into two ridges. Short ridges (or dots) are ridges that are significantly shorter than the average ridge length on the fingerprint. Minutiae and patterns are very important in the analysis of fingerprints since no two fingers are identical so far. To acquire a fingerprint as an image, a scanner system is required. In general, the fingerprint image is not saved; instead, it is converted into binary code which is used for verification. This binary code is created from the minutiae that are extracted from the fingerprint and known as the fingerprint template. Depending on the scanner manufacturer and the algorithms used, these templates can vary across devices. The algorithm cannot be used to re-convert the binary data to an image, so no one can duplicate your fingerprints.

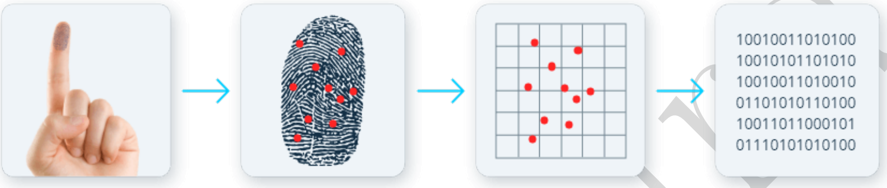

Figure 3. The process of extracting binary data from a fingerprint [9]

3. Literature review

Biometrics in cryptography has been an ongoing research topic, with many works being done in the field. In his paper, Sakre [16] introduces a novel method for exchanging symmetric keys based on personal fingerprint payloads using the SHA-1 hash algorithm. The key is then securely delivered to the other parties via an asymmetric cryptosystem. Barman et al. [5] introduce a key-exchange protocol using the biometric data of the sender and receiver. Users register their biometric information on a central server, which facilitates contact between registered users. Using a biometrics-based cryptographic construction, a user produces a cryptographic key at random and distributes it to another user. Both the sender and the receiver are guaranteed the confidentiality of the biometric information.

Wang et al. [20] propose a Diffie-Hellman key exchange and secret sharing-based fingerprint authentication method that protects user privacy. In order to securely distribute fragments of important private information among a distributed network or group, they use a secret sharing scheme, which lessens the workload on the template storage center (TSC) and the users. The user's original fingerprint template is kept in ciphertext format in TSC to ensure the security of template data. To further safeguard the privacy of the user's data, the DH key exchange protocol enables TSC and the user to encrypt the fingerprint template in each query using a unique one-time random key.

Juels and Sudan [12] proposed a cryptographic construction called fuzzy vault construct. The authors presented its application for a fingerprint-based security system, called fingerprint fuzzy vault. The fundamental idea is to conceal the cryptographic key in a list that has been jumbled and is made up of real fingerprint traits and made-up chaff features. The fuzzy vault's security is based on the difficulty of the polynomial reconstruction problem.

Ueshige and Sakurai [18] proposed a one-time authentication protocol that can create biometric authentication-based secure sessions. Both the fresh biometric data and the saved templates are subjected to a one-time transformation that is specific to the session. To verify the subject's authenticity, a comparison of the two altered templates is made. Bringer et al. [7] employed the Goldwasser-Micali cryptosystem for biometric authentication. With the help of this system, the biometric comparison can be done in an encrypted domain. The system ensures that the biometric data saved in the database cannot be explicitly linked to any user identity; instead, it just checks to see if the data associated with an identity is present. Barni et al. [6] proposed a scheme for privacy-preserving authentication based on fingerprints. The ElGamal cryptosystem, which enables biometric comparison in encrypted domains, is used in this technique. Upmanyu et al. [19] proposed a blind authentication protocol that is based on homomorphic encryption. The drawback of these authentication protocols is that they can only authenticate the subject, but they cannot produce the cryptographic keys required for secure communication.

The "Secure Ad-hoc Pairing with Biometrics: SAfE" protocol proposed by Buhan et al. [8] uses the fuzzy extractor scheme and can be used to establish a secure link between two parties. Unlike many biometrics-based protocols, this protocol does not involve a biometric template database or server. However, its drawback is that it shares the biometric data between the two parties and requires mutual trust among them. Additionally, a secure channel is needed for the exchange of biometric information.

4. Proposed methodology

The following Chapter explains how fingerprint data is extracted, and proposes two different approaches to utilizing said data in DH parameter generation. Before any key generation is attempted, it is necessary to extract the fingerprint data in a usable format. This process, and the size of the fingerprint template, can vary depending on the fingerprint module was used – FPM10A (Fig. 4).

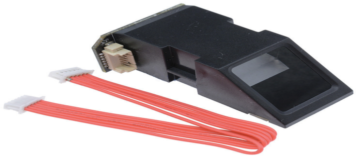

Figure 4. FPM10A fingerprint scanner

The module was connected to a simple Arduino setup to facilitate accessing the fingerprint templates in later steps.

This module (as well as many other scanners available on the market) is compatible with the Adafruit fingerprint scanner library. In the Adafruit library, fingerprints are converted into a 512 byte model, which is composed of 2 × 256 byte fingerprint templates. Upon finger enrollment, a user is asked to scan their fingerprint twice, which creates two templates that make up the model. In total, this is 4096 bits. This binary length is satisfactory for further parameter generation, as the recommended size of parameter p is at least 2048 bits [1]. Before continuing with the explanation of the proposed approach, it is worth noting that all features of the fingerprint binary string are handled by the Adafruit library, and may be different in other scanners (but the main ideas remain the same).

In terms of generable parameters, the values of the prime number p, generator g, and private keys a and b are chosen by the participating parties (p and g in agreement, a and b privately). Public keys A and B, as well as the shared secret x, are all calculated based on the initial parameters. For the research, two parties who wish to communicate - Alice and Bob - will be introduced. As there are no special constraints on private keys, except sufficient length, their private keys a and b will be calculated from their respective fingerprint templates. While a different generator g can be chosen, the best practice recommendation is to keep g at a low value for simplicity [13]. Therefore, g will be equal to 2. However, generating the prime p will not be as straightforward. According to best practices, it should be a prime number that is greater than the private keys of both parties. Generally, parameter p is agreed up prior to the key exchange, and any generated private keys will be bound by it. But, since the parties do not know the binary values of their fingerprint templates in advance, a special approach should be employed to make sure a large enough p is chosen of the remainder of the Diffie-Hellman algorithm. The paper proposes two different solutions.

Approach 1 – Generating a and b from biometrics, and keeping p as a random prime: In the first approach, fingerprint binary strings from Alice (a) and Bob (b) are used in full as their private keys, resulting in 4096-bit keys. The prime p will be a randomly generated prime number that is larger than both a and b. Since Alice and Bob obviously cannot disclose their private keys to agree on the value of prime p, they will each propose a candidate prime p that is greater than their private key. After exchanging the candidate primes, the larger prime will be chosen as p, as it is guaranteed to be larger than both private keys. Once p is selected, the Diffie-Hellman algorithm can proceed per usual rules. Step by step, the algorithms would work as follows:

1. Alice and Bob use their fingerprint templates as their private keys a and b.
2. Alice and Bob each generate a random *candidate prime* p_a / p_b in the following way: A bit string starting with 1 and ending with 1 is generated, of length 4096.

This is done to maximize the likelihood of generating a prime number since even number strings end with 0.

3. If the bit string is a number greater than the private key, proceed to the next step. If it is not, keep regenerating the bit string.
4. After the bit string is obtained, check if it is a *probable prime*, using Miller-Rabin [15] and Lucas-Lehmer [14] tests.
5. If it is a prime, proceed to the next step. If it is not, find the next probable prime higher than that number (using the same primary tests from the previous step).
6. After Alice and Bob generate their candidate primes p_a and p_b they exchange them, and the larger of the two becomes the prime p that will be used.
7. After the generation of p, all necessary parameters are obtained, and Diffe-Hellman can continue as usual. Figure 5 showcases a sample key exchange using this approach.

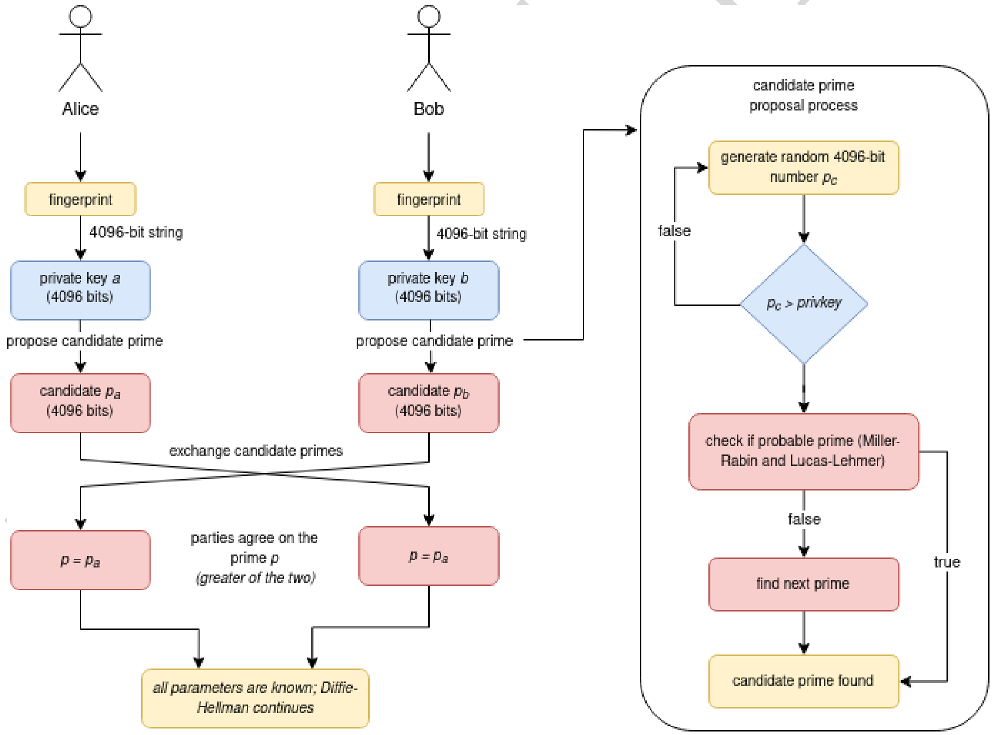
Alice
Bob
fingerprint
fingerprint
4096-bit string
4096-bit string
private key *a* (4096 bits)
private key *b* (4096 bits)
propose candidate prime
propose candidate prime
candidate p_a (4096 bits)
candidate p_b (4096 bits)
exchange candidate primes
$p = p_a$
parties agree on the prime p *(greater of the two)*
$p = p_a$
all parameters are known; Diffie-Hellman continues
candidate prime proposal process
generate random 4096-bit number p_c
false
p_c > privkey
check if probable prime (Miller-Rabin and Lucas-Lehmer)
false
true
find next prime
candidate prime found

Figure 5. Approach 1: Generating a and b from biometrics

The benefits of this approach are that the entire fingerprint template is used as the user's private key, and the resulting secret key is very large (4096 bits). On the other hand, prime generation at this scale is a computationally challenging task, and finding p may prove to be a slow process. Moreover, in this approach, biometrics are only used for the private keys. With these considerations in mind, the authors propose another solution that employs biometrics in both the private keys, and generation of prime p.

Approach 2 – Generating a, b and p from biometrics: While using all 4096 bits of a fingerprint template does result in stronger keys, research has shown that 2048-bit keys offer sufficient practical security. Therefore, if only 2048 bits of a fingerprint were used in the private key, the other half could be utilized in the generation of prime p. Naturally since p is a parameter that needs to be exchanged between the parties, it would not be acceptable to transmit raw and private fingerprint data. Instead, in this approach, the fingerprint would be randomly scrambled into chunks of 32 bits each, resulting in 128 chunks. Also, 64 random chunks (2048 bits) are then used as the private key (a or b). The other 2048 bits are used as the basis of the user's candidate prime pc - either as is (if the number is prime) or the next prime number is picked. The rest of the algorithm would function the same as Approach 1: Alice and Bob would exchange their candidate primes, the larger of them would be selected as p, and Diffie-Hellman could proceed. A diagram of this flow can be found in Figure 6.

The following paragraphs outlines the steps involved in this approach:

1. The fingerprint templates of Alice and Bob (respectively) are scrambled into 128 32-bit chunks.
2. Two-bit strings of 2048 bits each are constructed at both ends.
3. The lesser 2048-bit string is selected to be used as the private key a / b. The other (greater) 2048-bit string will be used as the basis of candidate prime p_a/p_b.
4. Check if the proposed bit string is a probable prime, using Miller-Rabin and Lucas-Lehmer tests.
5. If it is a prime, proceed to the next step. If it is not, find the next probable prime from that number (using the same primary tests from the previous step).
6. After Alice and Bob generate their candidate primes p_a and p_b they exchange them, and the larger of the two becomes the prime p that will be used.
7. After the generation of p, all necessary parameters are obtained, and Diffie-Hellman can continue as usual.

This approach works with smaller, albeit still secure, key values which can result in faster generation. Moreover, it maximizes the utilization of biometrics, using fingerprint data to produce three out of four generable parameters (everything except g, which is always agreed to be a small value).

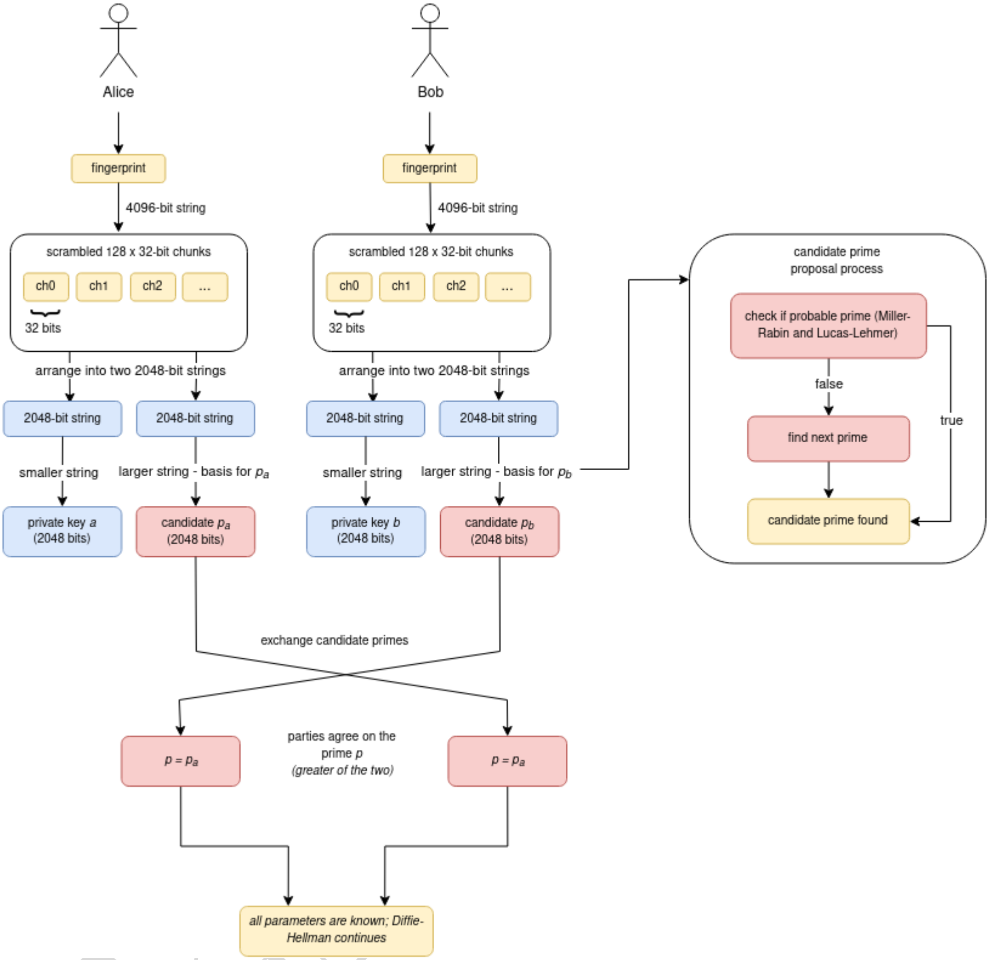
Alice
Bob
fingerprint
fingerprint
4096-bit string
4096-bit string
scrambled 128 x 32-bit chunks
ch0
ch1
ch2
...
32 bits
scrambled 128 x 32-bit chunks
ch0
ch1
ch2
...
32 bits
candidate prime proposal process
check if probable prime (Miller-Rabin and Lucas-Lehmer)
false
true
find next prime
candidate prime found
arrange into two 2048-bit strings
arrange into two 2048-bit strings
2048-bit string
2048-bit string
2048-bit string
2048-bit string
smaller string
larger string - basis for p_a
smaller string
larger string - basis for p_b
private key a (2048 bits)
candidate p_a (2048 bits)
private key b (2048 bits)
candidate p_b (2048 bits)
exchange candidate primes
$p = p_a$
parties agree on the prime p (greater of the two)
$p = p_a$
all parameters are known; Diffie-Hellman continues

Figure 6. Approach 2 – Generating a, b and p from biometrics

5. Results and discussion

Both approaches were experimentally implemented in Java, with the BigInteger library used for all mathematical operations. The example code has been made available on GitHub.

5.1. Experimental results

Table 1 below compares the runtimes of individual steps for each approach. The average of 10 runs is taken as the average time, and tests were executed on an AMD Ryzen 7 4800H CPU with 32 GBs of RAM.

Table 1
Time taken for different operations in both approaches

Operation	Approach 1 (ms)	Approach 2 (ms)
private key generation	0.85	5.85
candidate prime generation	5367.85	571.65
selection of prime p	0.02	0.02
public key generation	45.10	6.95
secret key generation	39.55	7.20
TOTAL	5453.37	591.67

As can be seen, the 2nd approach is more performant, averaging at around 9.22 times faster execution of Diffie-Hellman, compared to the 1st approach. All individual operations in the 2nd approach are also faster, except for private key generation. This is explained by the fact that the second algorithm needs to spend some time to scramble the fingerprint template data, and choose the correct 2048-bit string as the private key.

The main bottleneck in both algorithms is the generation of the candidate prime. Generation of primes at the scale required (4096 and 2048 bits) is a computationally expensive process, which leads to increased run times. Expectedly, the bottleneck is less disruptive in the 2nd approach, as it needs to generate a much smaller prime than the 1st approach. If the communicating parties wish to generate the shared secret key once, and conduct all future correspondence using the same key, slower generation times might not be an issue. However, nowadays Diffie-Hellman is most often used in its “ephemeral” variant. In ephemeral Diffie-Hellman, the key exchange is at “session-level”, with each new session using different starting parameters and resulting in a new secret key (the old keys are discarded). In that case, where real-time performance is a requirement, the 2nd approach might be preferred.

5.2. N-bit security

In terms of key length, both approaches offer appropriate security. The term n-bit security refers to the property of an algorithm in which an attacker would have to perform an average of 2n operations to break it [2]. Generally, attacks that take more than 2100 operations to break an algorithm are considered to be far too impractical to conduct [3]. For symmetric keys, their size should be at least 2n, whereas algorithms that rely on modular exponentiation / prime numbers should have much larger key sizes. According to NIST (National Institute of Standards and Technology) recommendations [4], to achieve 112-bit security (comparable to symmetric 3DES), the private key should comprise at least 224 bits, and p should be at least 2048 bits. Moreover, for 128-bit security (comparable to AES-128), key sizes should be 256 bits for the private key, and 3072 bits for p. It can be seen that both approaches satisfy 112-bit security. 224-bit security is satisfied by the 1st approach (due to using 4096-bit primes), but the 2nd approach falls slightly short due to its 2048-bit prime size.

This approach could potentially be modified to use 1024 bits for the private key, and the remaining 3072 for the prime p, thereby satisfying 128-bit security at the cost of increased computation time.

5.3. Key pool size

Another consideration is the key pool available in both approaches. If the entire fingerprint is used as the user's private key, the user has a pool of 10 available private keys. If a private key accidentally leaked or was hacked, it could not only compromise the user's conversations but potentially their identity as well. On the other hand, the 2nd approach uses scrambled data from a fingerprint, not the original bit string. This ensures that, even if a private key were to leak, the user's biometrics would still be protected, as it would not be possible to reconstruct the actual fingerprint template from available data. Moreover, since the private key uses 64 chunks of 32 bits (the other 64 chunks are used in the generation of prime p), the possible pool of keys is 128! / 64!, which is approximately 3.04 × 10126 possible permutations.

All in all, while both approaches are viable, the 2nd approach appears to have more practical benefits. While the 1st approach offers slightly better n-bit security, the 2nd approach is more computationally efficient, has a larger key pool and offers more precautions against user biometrics leaking.

6. Conclusion

To summarize, the result of this research paper is the proposal of two different algorithms for the generation of Diffie-Hellman key exchange parameters, based on fingerprint biometrics. Initially, the fingerprint templates are extracted as 4096-bit strings using an Adafruit-compatible fingerprint scanner. Afterward, two approaches are described. In the first approach, fingerprint templates are used as the users' private keys, and the p is a random prime number found to be greater than both fingerprint bit strings. In the second approach, the fingerprint templates are scrambled into 128 chunks of 32 bits each. These chunks are then assembled into 2048-bit bit strings, one of which is used as the private key, and the other as the basis for the generation of prime p. The prototypes for both algorithms are developed in Java, showing them to be feasible. After computational and security analyses, the second approach was found to be more advantageous, offering increased performance and a greater key pool over the first approach.

In the future, the authors plan to improve upon the second proposed approach and come up with a way to speed up prime generation, making the algorithm more usable in practice. The authors believe this paper presents a viable case for the usage of fingerprint biometrics in Diffie-Hellman that warrants additional future consideration and experimentation.

References

[1] Adrian D.: *Imperfect forward secrecy*, ACM SIGSAC Conference on Computer and Communications Security, 2015.

[2] Alwen J.: *The bit-security of cryptographic primitives*, AWS Wickr. https://wickr.com/the-bit-security-of-cryptographic-primitives-2/. [accessed Sep. 13, 2023].

[3] Aumasson J.P.: *Too Much Cryp*, Real World Crypto, 2020.

[4] Barker E.: *Recommendation for key management*, NIST: National Institute of Standards and Technology, 2019.

[5] Barman S., Chattopadhyay S., Samanta D., Panchal G.: A novel secure key-exchange protocol using biometrics of the sender and receiver, *Computers and Electrical Engineering*, vol. 64, pp. 65 – 82, 2017.

[6] Barni M.: *Privacy-preserving fingercode authentication*, ACM workshop on Multimedia and security, 2010.

[7] Bringer J.: An application of the goldwasser-micali cryptosystem to biometric authentication, *Information Security and Privacy*, vol. 7, pp. 96 – 106, 2007. doi: 10.1007/978-3-540-73458-1_8.

[8] Buhan I.: *Cryptographic keys from NOISY DATA: Theory and applications*, PS University Press, 2008.

[9] Durairajan M.S., Saravanan R.: Biometrics Based Key Generation using Diffie Hellman Key Exchange for Enhanced Security Mechanism, *Recent Trends in Biotechnology and Chemical Engineering*, vol. 6(9), pp. 4359–4365, 2014.

[10] Gillis A.S.: *What is Diffie-Hellman key exchange?: TechTarget*, Security. https://www.techtarget.com/searchsecurity/definition/Diffie-Hellman-key-exchange.

[11] Ha F., Anderson R., Daugman J.: Combining crypto with biometrics effectively, *IEEE Transactions on Computers*, vol. 55(9), pp. 1081–1088, 2006.

[12] Juels A., Sudan M.: A fuzzy vault scheme, *Designs, Codes and Cryptography*, vol. 38(2), pp. 237 – 257, 2006. doi: 10.1007/s10623-005-6343-z.

[13] Kojo M., Kivinen T.: *RFC 3526: More modular exponential (MODP) diffie-hellman groups for Internet Key Exchange (IKE)*, IETF Datatracker, accessed Sep 13, 2023.

[14] Lucas-Lehmer Test, Prime. https://www.rieselprime.de/ziki/Lucas-Lehmer_test. [accessed Sep. 13, 2023].

[15] Primality tests – Number Theory. https://crypto.stanford.edu/pbc/notes/numbertheory/millerrabin.html. [accessed Sep. 13, 2023].

[16] Sakre M.I.: Exchanging Biometric Keys in Secrecy, *International Journal of Scientific and Engineering Research*, vol. 6(9), pp. 1113–1120, 2015.

[17] Socheat S., Wang T.: Fingerprint enhancement, minutiae extraction and matching techniques, *Journal of Computer and Communications*, vol. 8(5), pp. 55 – 74, 2020.

[18] Ueshige Y., Sakurai K.: A proposal of one-time biometric authentication, *Proceedings of The 2006 International Conference on Security and Management*, 2006.

[19] Upmanyu M., Namboodiri A.M., Srinathan K., Jawahar C.V.: Blind authentication: A secure crypto-biometric verification protocol, *IEEE Transactions on Information Forensics and Security*, vol. 5(2), pp. 255 – 268, 2010.

[20] Wang H., Luo M., Ding Y.: Privacy-preserving fingerprint authentication using D-H key exchange and secret sharing, *Security and Communication Networks*, vol. 21, pp. 1 – 12, 2021. doi: 10.1155/2021/5344696.

Affiliations

Aldin Kovačević
International Burch University Ilidža, Department of Information Technologies, Bosnia and Herzegovina, aldin.kovacevic@ibu.edu.ba

Muzafer Saračević
University of Novi Pazar, Department of Computer Sciences Novi Pazar, Serbia, muzafer.saracevic@uninp.edu.rs

Amor Hasić
University of Novi Pazar, amorhasic@gmail.com

Received: 31.01.2024
Revised: 15.04.2024
Accepted: 15.04.2024