End-to-End Encrypted Messaging Based on PGP with Forward Secrecy

Yiming Shen1*
1 College of Information Science and Engineering, Ocean University of China, Qingdao, Shandong, 266100, China
*Corresponding author’s e-mail: shenyiming@stu.ouc.edu.cn

Abstract. In the era of big data, more and more users prefer messaging applications with end-to-end encryption to protect their privacy. However, if a hacker gets the password or encrypting key, it is possible to login into the account, or intercept the network packets and get the original messages. This paper introduces a new decentralized online messaging system that combines PGP and Diffie–Hellman key exchange and gives forward secrecy to end-to-end encrypted messaging. It is not possible for the hacker to decrypt the intercepted messages once the temporary session is ended, even if the hacker gets the PGP private keys.

1. Introduction
In the era of big data, security and privacy are more and more important. Widely used applications such as Facebook Messenger, LINE, and WhatsApp have adopted end-to-end encrypting technologies, which makes the server or any third-party adversary that intercepts data cannot read the message content[1]. Also, a lot of Email clients now support encrypting emails with PGP/GPG and make emailing more secure[8]. However, if the password or key is leaked, restoring the original message content is possible[11]. We argue that encryption should not only be used to hide the content of the conversation, but also to provide forward secrecy against future compromises. In addition, we should use an authentication mechanism to confirm the identity of the other end of the conversation. Furthermore, these should all be decentralized to reduce dependence on third-party services.

This paper introduces a new end-to-end encrypted online messaging application design, which is quite simple and straightforward. The idea is to combine PGP and Diffie–Hellman key exchange. Both of these two technologies are widely used and reliable. PGP is valuable for end-to-end encrypted messaging, as people can import public keys only that they trust, and establish a decentralized web of trust. As a result, is designed to be the base of this protocol. By encrypting and signing all data in the network packets, PGP can give us confidentiality and integrity. And with Diffie–Hellman key exchange, a new ephemeral key pair will be generated for each temporary session, used for encrypting messages, and discarded at the end of it, which gives us forward secrecy. Users can trust other users by importing their public keys and chat with each other with this application. A third-party adversary may intercept network packets, but decrypting the intercepted network packets and getting the original message content is not possible after the end of each temporary session, even if the adversary gets all private keys.

This article first introduces background information about technologies and algorithms that will be used in this system, including PGP, Diffie–Hellman key exchange, and forward secrecy. And then, the proposed protocol of the messaging system will be discussed. Based on this protocol, a simple
demonstration program was implemented and will be introduced. After that, the limitations of this project will be discussed. And finally, we review some related work and then draw some conclusions.

2. Background

2.1. PGP
PGP (Pretty Good Privacy) is an encryption program that provides cryptographic privacy and authentication for communication. It was created to add end-to-end encryption capabilities to email by Phil Zimmerman in 1991[2]. PGP can be used for encrypting and signing messages such as texts, emails, and files. PGP has become an IETF standard, the OpenPGP standard. GPG (GnuGPG) is a free software implementation of the OpenPGP standardized specification, and it is the most widely used implementation. As a result, PGP usually means GPG instead of the original PGP created by Phil Zimmerman.

In contrast to things like PKI that is based on a centralized trust model, PGP is based on a decentralized trust model, the web of trust[2]. There is not a centralized controller (e.g., a CA), users accept and sign keys of each other, and the more people sign a public key, the more trustable that public key is. Key signing parties are a way to strengthen the web of trust, which is an event at which people show each other their PGP public keys and then use their private keys to digitally sign and publish other people’s public keys.

2.2. Diffie–Hellman Key Exchange
Diffie–Hellman key exchange is an algorithm published in 1976 by Diffie and Hellman, and it can be used to securely exchange cryptographic keys over a public channel[3]. It is widely used to provide forward secrecy by generating new key pairs for each session and discarding them at the end of the session.

The Diffie-Hellman key exchange itself does not provide authentication for both sides of the communication, so it is vulnerable to man-in-the-middle attacks. An active attacker is able to perform two Diffie-Hellman key exchanges in the middle of the channel, one with Alice and the other with Bob. The attacker is able to pretend to Alice that he is Bob, and vice versa, and decrypt messages of any person and re-encrypt them and pass it on to another person. Therefore, a mechanism that can verify the identity of both communicating parties is usually needed to prevent such attacks.

2.3. Forward Secrecy
In cryptography, forward secrecy (FS), sometimes referred to as perfect forward secrecy (PFS), is a security feature provided by many protocols that the compromise of a long-term key does not lead to the compromise of past session keys[4]. Forward secrecy protects communications in the past from the threat of future compromise of passwords or keys. If a system has forward secrecy, it can guarantee the security of historical communications in the event of private key compromise, even if the system is actively attacked. To provide perfect forward secrecy, we usually use asymmetric algorithms such as the well-known Diffie-Hellman key exchange protocol.

3. Design

3.1. The Proposed Protocol
As a precondition, let us assume that both sides, say, Alice and Bob, have imported the public key of each other securely. This can be done face to face at key signing parties. And as for the protocol, it is pretty simple and straightforward. The bottom layer of this protocol is based on PGP, specifically, all data in the network packets to the receiver will be encrypted and signed by the sender. Above that layer, handshake and ciphertext data will be transmitted. When anyone of Alice and Bob wants to communicate with the other one, a Diffie–Hellman key exchange will be done first, and an ephemeral session key will be shared. The subsequent messages will be encrypted with this key using symmetric
encryption algorithms. To prevent replay attacks, timestamps and randomly initialized sequence numbers are added to handshake packets and message packets.

This protocol can be deployed above any common protocols, and for practical reasons, it is recommended to connect to a WebSocket Secure server to forward network packets. Anyone may build his own forwarding server, so decentralized end-to-end encrypted messaging is possible. Messaging on a LAN without an Internet connection is also possible.

3.2. Implementation

Based on the introduced protocol, a simple demonstration program was implemented. Considering that direct connections between clients are not always possible, clients do not connect directly, on the contrary, there is a WebSocket Secure server for clients to connect directly, designed to forward network packets to each other. As mentioned above, anyone may build his own public or private forwarding server, so decentralized end-to-end encrypted messaging is possible.

The client is based on the B/S model, that is to say, there is also an HTTP web server on the client-side. The web server is based on the Flask framework, which is a lightweight WSGI web application framework written in Python. And the webserver is in charge of invoking GPG using the GPGME (GnuPG Made Easy) library. The user interface is in the web browser, as a result, it is a cross-platform client. The browser is responsible for network connections, users would input the address of any forwarding server and connect to it first.

![Figure 1. The browser-based user interface](image)

3.3. Security Analysis

In this part, we analyze the security of our proposed protocol. Actually, PGP and Diffie-Hellman key exchange are complementary. In the traditional PGP usage scenario, an attacker who gets the private key is possible to decrypt the intercepted network packets and get the original messages. We embed the Diffie-Hellman key exchange in PGP to solve the previous problem. On the other hand, PGP can be a protective layer to protect Diffie-Hellman key exchange from man-in-the-middle attacks. The specific analysis is as follows.

We first analyze the confidentiality and completeness of the end-to-end encrypted messaging in this proposed protocol. We can easily get the following theorem.

**Theorem 1.** The message of end-to-end encrypted in this proposed protocol is confidential and complete based on PGP.
Proof. According to the properties of PGP[7], we can know that PGP is a combination of conventional cryptography and public-key cryptography. The authentication of PGP ensures the completeness of the end-to-end encrypted messaging and the confidentiality of PGP actualizes the confidentiality of our proposed protocol.

Theorem 1 implies that our protocol can resist man-in-the-middle attacks. Suppose the man-in-the-middle intercepts and stores all the packets sent by both communicating parties in a certain session and obtains the private keys of both communicating parties in the future. The man-in-the-middle can decrypt the PGP packets, but he can only see the DH handshake process and cannot get the shared keys of the communicating parties, so he cannot perform subsequent decryption.

Then we analyze the forward secrecy of the end-to-end messaging in this proposed protocol.

In our protocol, when anyone of Alice and Bob wants to communicate with the other one, a Diffie–Hellman key exchange will be done first, and an ephemeral session key will be shared. Due to the session key is ephemeral, even if the attacker gets the long-term PGP private key, he cannot recover any session key, which ensures the forward secrecy.

Additionally, considering replay attacks, if the client receives a packet with illegal sequence number or timestamp, it will discard the message and warn the user. At the same time, attackers cannot forge packets arbitrarily, so replay attacks are avoided.

4. Limitations

The Diffie–Hellman key exchange must be done first to create a temporary session. Meanwhile, the forwarding server does not store any user data. As a result, both of the two users need to be online whenever a new session is created. Otherwise, the initial handshake could not be done, and no message could be sent. So as usual, users may have to compromise between ease of use and privacy.

This is just a simple protocol and demonstrating implementation that only support one-on-one chat, but not support group conversations. Extending this protocol without reducing security to support group conversations is possible.

5. Related Work

PGP is widely used to protect security and privacy. It is one of the choices to encrypt emails[8]. Sachin et al.[9] proposes two modifications of PGP to improve the behavior and speed of it when processing group emails. Didit et al.[7] applied PGP in eGovernment applications as an application message security by implementing five major components of PGP. Yusuf et al.[10] created an application named Mini PGP as a simplified version of PGP, and it is more secured against dictionary attack and spyware if compared with the conventional one.

There are some studies that improved existing systems and give forward security to them. A common way to encrypt emails is to generate a short-term symmetric key, encrypt the email with this short-term key, and then encrypt this short-term key using the recipient’s PGP public key and transmit them together. Hung-Min et al.[5] argue that this protection cannot provide perfect forward secrecy, because once the recipient’s private key is compromised, all previously used temporary keys will also be compromised, resulting in email content leakage. They propose two new email protocols that provide perfect forward secrecy and are flexible and suitable to the email system. It has long been common for people to use programs like PGP that use long-term keys to provide confidentiality and authenticity. Nikita et al.[6] argue that in everyday chat, most communications should have just the opposite of the above two properties, perfect forward secrecy, and repudiability. They propose a protocol named “off-the-record messaging” which is more suitable for a daily chat. They also implemented this protocol as an extension to the Linux messaging client GAIM.

6. Conclusion

Forward secrecy is generally recognized as a desirable feature, it helps people protect their privacy even if long-term secret keys are compromised. And PGP is a reliable technology that is widely used.
In this paper, we combine these technologies and algorithms and give forward secrecy to PGP-based end-to-end encrypted messaging.

We have developed an end-to-end encrypted messaging protocol, which gives users confidentiality and authenticity assurances, and at the same time, allows users to communicate online in a perfect forward secret manner. Users could take full advantage of the existing web of trust and have a decentralized online messaging experience.

We have implemented this protocol as a client based on the B/S model, which has the ability to run on any operating system that supports Python and has a browser installed. We care about privacy and security, and our hope is to give more choices to people to have private and secure conversations on the Internet.

Acknowledgments
I wish to thank Professor Qu, my supervisor, for his instructive advice and guidance. I also would like to thank my friends and my fellow classmates for their useful help and comments.

References
[1] Ermoshina, K., Musiani, F., Halpin, H. (2016) End-to-end encrypted messaging protocols: An overview. In: International Conference on Internet Science. Springer, Cham. pp. 244-254.
[2] Garfinkel, S. (1994) PGP: Pretty Good Privacy. O'Reilly Media, California.
[3] Diffie, W., Hellman, M. (1976) New directions in cryptography. IEEE transactions on Information Theory, 22(6): 644-654.
[4] Günther, C.G. (1989) An identity-based key-exchange protocol. In: Workshop on the Theory and Application of Cryptographic Techniques. Springer, Berlin, Heidelberg. pp. 29-37.
[5] Sun, H.M., Hsieh, B.T., Hwang, H.J. (2005) Secure e-mail protocols providing perfect forward secrecy. IEEE Communications Letters, 9(1): 58-60.
[6] Borisov, N., Goldberg, I., Brewer, E. (2004) Off-the-record communication, or, why not to use PGP. In: ACM workshop on Privacy in the electronic society. Washington. pp. 77-84.
[7] Suprihanto, D., Priyambodo, T.K. (2017) The Implementation of Pretty Good Privacy in eGovernment Applications (Case Study on the Official Scripts Electronic Applications in Bantul). International Journal of Information Engineering and Electronic Business, 9(4): 1-6.
[8] Elkins, M. (1996) MIME security with pretty good privacy (PGP). RFC 2015.
[9] Tripathi, S., Biswas, G.P. (2010) Secure E-Mail Messaging to Selected Group Members Using PGP Technique. International Journal of Computer Applications, 975, 8887: 76-80.
[10] Kurniawan, Y., Albone, A., Rahyuwibowo, H. (2011) The design of mini PGP security. In: Proceedings of the 2011 International Conference on Electrical Engineering and Informatics. Bandung. pp. 1-4.
[11] Espinoza, A.M., Tolley, W.J., Crandall, J.R., Crete-Nishihata, M., Hilts, A. (2017) Alice and bob, who are they?: Analysis of end-to-end encryption in the LINE messaging application. In: 7th USENIX Workshop on Free and Open Communications on the Internet. Vancouver. pp. 1-9.