Key Exchange Using an Online Trusted Third Party

ALEXANDER HAAR, Universität Kassel, Deutschland

Abstract
The Trusted Thid Party (TTP) is an effective key exchange protocol for a small group of users. If the TTP plays an active role in all exchanges, it can only be build with symmetric ciphers. In addtition the TTP can provide secure channel bindings. The TTP is a statically secure exchange protocol, but on the other side it lacks Perfect Forward Secrecy (PFS).

CCS Concepts: • **Security and privacy** → Block and stream ciphers.

Additional Key Words and Phrases: key exchange, online, trusted third party

1 INTRODUCTION

In our modern society, the internet plays an integral role, serving as an indispensable tool in our daily lives. Whether working remotely, conducting financial transactions, or connecting with friends through chat, we rely on its seamless functionality. Safeguarding our online activities is paramount, necessitating the implementation of various protocols to enhance security. Among these protocols, the Online TTP [1] stands out. This protocol facilitates the secure exchange of keys over the internet, ensuring our communications remain private and protected.

2 OVERVIEW

To ensure a secure key exchange utilizing symmetric ciphers exclusively, the TTP takes on an active role during both registration and the actual key exchange. Consequently, users engaging in this key exchange protocol must have implicit trust in its mechanisms. This trust forms the basis for labeling the protocol as a trusted third party. The TTP which is actively involved in this process, is referred to as a key distribution center. The TTP has the capability to offer secure channel bindings [1] as necessary.

This protocol proves to be efficient for clients, requiring them only to verify a message and decrypt the ciphertext, minimizing their tasks. However, the TTP bears a significant workload. It is responsible for generating private and session keys and maintaining communication with users involved in the key exchange, as detailed in the upcoming chapter. This heavy load on the TTP becomes pronounced when multiple users engage in simultaneous key exchanges, leading to scalability challenges. Moreover, establishing a scalable and reliable system is costly. Another drawback arises when the protocol is compromised, all past and future key exchanges become vulnerable.

Due to these scalability and security concerns, the system is impractical for internet use, where a large number of users are expected. Instead, the TTP is best suited for scenarios with a smaller user group, such as a corporate network.

3 KEY EXCHANGE PROTOCOL

In this section, we'll delve into the key exchange protocol, which is conveniently divided into two parts. Initially, a new user initiates the process by registering with the TTP and obtains their private key. Following this registration, the user gains the ability to initiate secure key exchange with other users.

3.1 Registration

This aspect of the protocol is crucial for the subsequent key exchange explanation, as the user requires their private key to encrypt the ciphertext.

Initially, a new user is required to transmit their identity to the TTP as a means of introduction, enabling the TTP to recognize the user. Once the TTP receives the user's identity, it proceeds to generate a random long-term secret key, utilizing both the cipher key space and the Message Authentication Code (MAC) key space.

To facilitate future identification of the user, the TTP maintains a table associating the identity with the private key. For security considerations, the actual key is not stored in plaintext; instead, only the user's identity is retained. Given the TTP's active role and the need to access the private key, a master password is employed. The TTP can generate the private key by inputting the user's identity and its master password into a function, yielding the corresponding private key.

Subsequently, the generated secret key is transmitted back to the user, marking the completion of the registration protocol. With this, the user is now prepared to engage in a key exchange.

3.2 Key Exchange

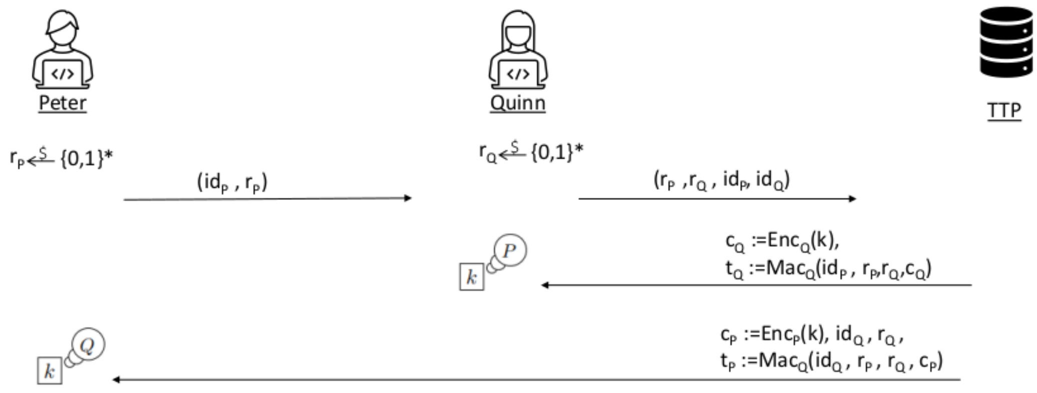
Peter
Quinn
TTP
$r_P \overset{\$}{\leftarrow} \{0,1\}^*$
(id_P , r_P)
$r_Q \overset{\$}{\leftarrow} \{0,1\}^*$
(r_P , r_Q , id_P, id_Q)
$c_Q := Enc_Q(k),$
$t_Q := Mac_Q(id_P , r_P, r_Q, c_Q)$
P
k
$c_P := Enc_P(k), id_Q , r_Q ,$
$t_P := Mac_Q(id_Q , r_P , r_Q , c_P)$
Q
k

Fig. 1. Key exchange between P and Q, Comparison [1]

In Figure 1 the key exchange is visualized. For this aspect of the protocol, we need two users, Peter (P) and Quinn (Q), that are listed inside the TTP table. First P computes a random large number $r_P \overset{R}{\leftarrow} R$. After generation, P transmits both his identity and r_P to Q.

Subsequently, Q generates a random large number $r_Q \overset{R}{\leftarrow} R$. Q proceeds to send both identities and the respective large numbers to the TTP. In the background, the TTP checks its table to verify if both users, P and Q, are listed and possess secret keys. If both users are identified in the table with secret keys, the TTP continues; otherwise, the key exchange process is halted.

After successful verification, the TTP initiates the generation of a session key from the key space, denoted as $k \overset{R}{\leftarrow} K$. Additionally, the TTP creates the following cryptographic elements:

$$c_Q \overset{R}{\leftarrow} \mathrm{Enc}_Q(k), \quad t_Q \overset{R}{\leftarrow} \mathrm{Mac}_Q(id_P, r_P, r_Q, c_Q)$$
$$c_P \overset{R}{\leftarrow} \mathrm{Enc}_P(k), \quad t_P \overset{R}{\leftarrow} \mathrm{Mac}_P(id_Q, r_P, r_Q, c_P)$$

Here, c_Q represents the encryption of the session key k using Q's private key, while t_Q is a valid MAC tag on the message (id_P, r_P, r_Q, c_Q). Similarly, for user P, the TTP generates analogous values using P's private key instead of Q's private key.

Following this, the TTP forwards (c_Q, t_Q) back to user Q, while user P receives (c_P, id_Q, r_Q, t_P). In this exchange, P additionally obtains the values id_Q and r_Q, allowing P to identify the communication partner. Subsequently, both users independently verify the validity of their respective MAC tags t. A successful verification allows the exchange to proceed, while an invalid tag prompts termination.

The final step involves decrypting the ciphertext to extract the session key k and confirming its membership in the key space. At this point, both users possess the session key k and are aware of the identity of their communication partner.

4 INSECURE VARIATIONS

In this section, we will explore four insecure variations by removing one element from the MAC tag in each case. Each insecure variation is susceptible to a specific attack that compromises the integrity of our protocol.

4.1 Key Exposure Attack

In this particular variation, we eliminate c_P from the MAC tag t_P. Here, an adversary begins by registering a new user R. Subsequently, the adversary initiates a conversation with P and after the exchange, both parties obtain a session key k'.

The adversary patiently waits until P and Q engage in an exchange. Upon interception of the message from the TTP to P, the adversary modifies c_P to c'_P. P is unable to verify the validity of c'_P, because it is erased from t_P. After P decrypts c'_P, he assumes it to be the session key. However, this k' is the session key from the previous exchange that the adversary is aware of. In result, all messages encrypted with k' can now be decrypted by the adversary.

A similar attack can be executed on Q. The strategy involves replacing c_Q with c'_Q. Upon decrypting it, Q obtains k', but cannot verify if c'_Q is valid. This leaves Q susceptible to the same security breach.

4.2 Replay Attack

In the second variation, we remove r_P from the MAC tag t_P. In this scenario, the adversary strategically records a key exchange for later use. Let's assume P sends (id_P, r'_P) to Q. Subsequently, the TTP responds to Q and sends a message (c'_P, t'_P, id_Q, r'_Q) to P.

The adversary patiently waits until the user initiates a new key exchange. At that point, the adversary intercepts the message from P to Q, discards the message and instead sends back (c'_P, id_Q, r'_Q, t'_P). As a result, P believes he is communicating with Q using a session key k, but unbeknownst to P, he is communicating with Q using the old session key k' from the previous conversation.

A corresponding attack can be executed on user Q. In this case, the adversary intercepts the message from the TTP to Q and substitutes c_Q with c'_Q. Q decrypts c'_Q and obtains k', the outdated session key.

4.3 Identity Missbinding Attack

In the third variation, we eliminate id_Q from the MAC tag t_P. Here, the adversary initiates the attack by registering a new user R. Subsequently, the adversary patiently awaits a key exchange between P and Q. Upon interception of the message sent from the TTP to P, the adversary alters id_Q to id_R. P, in turn, is unable to verify the validity of the identity. Despite P and Q sharing the same key, the misbinding occurs as P erroneously believes he is conversing with R.

A parallel misbinding attack can be executed on Q. In this case, the adversary intercepts the first message from P to Q, modifying id_P to id_R. On the subsequent message from Q to the TTP, the adversary intercepts again, altering id_R back to id_P. Once the protocol is completed, Q incorrectly perceives she is communicating with R, while P mistakenly believes he is engaged in a conversation with Q. The misbinding of the parties persists.

4.4 Secure Channel Bbindings Attack

In the final variation, we omit r_Q from the MAC tag t_P. During a key exchange between P and Q, when the TTP sends a message to P, the adversary intercepts the message and alters r_Q to 0. Despite P and Q agreeing on a session key k, the adversary remains unaware of this key. The significance lies in the fact that P and Q now hold conflicting channel bindings. Consequently, the key exchange fails to reach a successful completion.

5 CONCLUSION

The TTP proves to be a valuable key exchange protocol, particularly in scenarios involving a small group of users. However, its efficiency diminishes in larger user groups due to the substantial workload placed on the TTP. Despite this limitation, the protocol's reliance on symmetric ciphers facilitates straightforward implementation. Additionally, it stands out as a statically secure key exchange protocol [1], assuming the underlying cipher is CPA secure and the MAC system is secure. Furthermore, the TTP can offer secure channel bindings as needed.

Nevertheless, the protocol is not PFS secure [2]. In the event that an adversary gains access to either P's key, Q's key, or the TTP's master key, all past and future conversations between P and Q become vulnerable to decryption. This limitation underscores the importance of considering alternative protocols in situations where PFS is a critical security requirement.

REFERENCES

[1] Dan Boneh and Victor Shoup. 2023. *A Graduate Course in Applied Cryptography*. Vol. 0.6. Stanford, California.

[2] Wikipedia. 2024. *Forward secrecy*. Retrieved February 19, 2024 from https://en.wikipedia.org/wiki/Forward_secrecy