CRYSTALS Kyber Integration into TLS

Stephan Müller

2023-07-14

Abstract

This document outlines the shortcomings of a current proposal to add CRYSTALS Kyber into TLS and proposes an alternative solution. As the key agreement operations in the SSHv2 and IKEv2 protocols are very similar to TLS, the proposal can be adopted for those protocols as well.

Contents

List of Figures

List of Tables

1 Introduction

Post-quantum computing cryptographic algorithms are designed and available for use. Considering that the Kyber algorithm is going to be mandated by US authorities in the future as a complete replacement for asymmetric key exchange and agreement, a proposal integrating Kyber into TLS is specified.

This proposal, however, has one central shortcoming: only the TLS server contributes to the security strength of the shared secret generated by Kyber. This shortcoming can be solved with a slightly improved approach where the client and the server both independent of each other contribute to the security of the communication channel where the channel even retains its security when one side has insufficient entropy.

Furthermore, the presented solution has an intrinsic remote peer authentication which may allow to dispense with subsequent authentication operations based on signatures. This authentication is intrinsic to the Kyber operation at a similar level as authentication is part of an AEAD symmetric algorithm.

1.1 Kyber Operation

Before outlining the shortcomings of the Kyber TLS integration proposal, a brief sketch of how Kyber works is relevant. This sketch is not meant to provide a mathematical explanation of the Kyber mechanism, but highlights the items relevant for this discussion. The Kyber mechanism concept is based on the following steps:

1. The Kyber encryption operation takes a public key `pk` which is used to encrypt a random number. The resulting ciphertext `ct` is returned. In addition, the encryption operation also generates a shared secret `ss` from the random number. The encryption operation performs the following steps:

   a. Generate a random number of 256 bits in size.

   b. Perform the Kyber INDCPA encryption operation using the random number from a, and the public key `pk` to generate the ciphertext `ct`.

   c. Calculate a hash of the random number from step a and the ciphertext `ct`.

   d. Process the hash from step c with SHAKE256 to generate the shared secret `ss`.

2. The Kyber decryption operation receives the ciphertext and uses the associated secret key `sk` to obtain the data and transform it into a shared secret `ss`. The following steps are performed for the decryption operation:
   a. Perform the Kyber INDCPA decryption operation using the ciphertext `ct` and the secret key `sk`.
   b. Perform an additional Kyber operation using the decrypted data from step a and parts of the secret key `sk`.
   c. Calculate the shared secret `ss` using SHAKE256 out of the result of step b.

The outlined mechanism is referenced as the Kyber Key Encryption Mechanism (Kyber KEM). The actual Kyber INDCPA operation is of no relevance to the discussion here, however, and is not further discussed.

1.2 Kyber TLS Integration Proposal

The proposal for integrating Kyber into TLS mandates the following operational steps:

1. The TLS client generates an ephemeral Kyber keypair `pk` and `sk`. The public key `pk` is transmitted to the TLS server.
2. The TLS server performs the Kyber encryption operation to generate the shared secret `ss` and the ciphertext `ct`. The ciphertext `ct` is returned to the client.
3. The client performs the Kyber decryption operation to generate the shared secret `ss`.

The shared secret `ss` on the server and client side are handed over to the regular TLS PRF to generate all required keys for the symmetric and authentication algorithm instances.

1.3 Shortcomings of the Kyber TLS Integration Proposal

When considering the proposal in light of the Kyber operation, the following is visible:

1. The client generates a random Kyber key pair. The public key `pk`, however, is transmitted unprotected over insecure networks and thus is considered to not contain any entropy. At most, it is usable to "mix" the shared secret `ss` a bit more.
2. Only the Kyber encryption operation gathers random numbers which remain private and protected. Thus, only the encryption operation provides the strength of the resulting shared secret. If the random number generated for the encryption operation is weak, an attacker may sniff the `pk` sent over the wire and "guess" the random number to obtain the shared secret

> ss. As the Kyber encryption is solely performed by the TLS server, the security of the entire TLS connection hinges solely on the security of the TLS server. The TLS client cannot contribute to the security strength of the shared secret at all.

The outlined issue may be argued to not be so much different compared to the use of Diffie-Hellman - where the shared secret is in jeopardy if one side has a weak random number generator - or the use of RSA-based keywrapping - which conceptually is not too much different from Kyber KEM by wrapping a shared secret.

2 Proposal to Strengthen TLS

Considering the current status quo, is it truly necessary that we remain at the current security state in case different solutions are present?

The CRYSTALS Kyber authors have specified the Kyber KEM algorithm upon which the challenged proposal rests on. These authors have provided a reference implementation at [1]. However, this reference implementation contains a gem which is not fully documented and specified in their documents: the Kyber Key Exchange Mechanism (Kyber KEX). This Kyber KEX algorithm is fully specified and documented by the author of this proposal at [2].

The Kyber KEX uses multiple Kyber KEM operations such that either side of the communication channel performs a Kyber encryption operation and thus contribute with their random numbers to the security strength of the communication link. Thus, if one side turns out to have weak random numbers, the communication link is still cryptographically strong.

Yet, the integration of the Kyber KEX into existing protocols may pose a challenge as a number of steps need to be performed as outlined in [2]. However, by properly adjusting the different steps, the Kyber KEX can be completed with just 2 network exchanges. With such a limited number of network exchanges, Kyber KEX should be included into the existing TLS handshake. The following section outlines the proposal to squeeze Kyber KEX into 2 network transmissions.

2.1 Kyber KEX Key Agreement

This section describes the Kyber KEX algorithm with bi-directional authentication. The proposal for unilateral authentication is given in section Unilateral Authentication.

2.1.1 Kyber KEX - Basic Concept

The following table outlines the general stages of an bi-directional authenticated Kyber Key Exchange (KEX) which are also all followed by the secure connection implementation.

Table 1: Basic Kyber Key Exchange Steps

Step	Alice (Inititiator)	Bob (Responder)
1	Key gen: `pk_i`, `sk_i`	Key gen: `pk_r`, `sk_r`
2	Send public key `pk_i` ;	Send public key `pk_r` ;
	Receipt of `pk_r`	Receipt of `pk_i`
3	Initiate key exchange - generate: ephemeral public key `pk_e_i`, ephemeral ciphertext `ct_e_i`, KEM shared secret `tk`, ephemeral secret key `sk_e`.	
4	Send KEX data `pk_e_i`, `ct_e_i`	Receipt of `pk_e_i`, `ct_e_i`
5		Calculate: ephemeral ciphertext `ct_e_r_1`, ephemeral ciphertext `ct_e_r_2`, Shared Secret `ss`
6	Receipt of `ct_e_r_1`, `ct_e_r_2`	Send `ct_e_r_1`, `ct_e_r_2`
7	Calculate: Shared Secret `ss`	

Note that steps 1 and 2 are to be performed once to generate a set of static key pairs and to securely exchange the public keys which is therefore marked as gray as it is not directly part of a given Kyber KEX operation.

The basic Kyber KEX steps are compressed such that only two network exchanges are required to perform a Kyber KEX handshake. By combining several steps into one, the 2-way handshake for a Kyber KEX operation is achieved which is illustrated with Figure [1].

2.1.2 Prerequisites

The following operations must be performed *before* any handshake operation commences. These operations are therefore outside of this protocol specification:

- Exchanging the Kyber static public keys `pk_i` and `pk_r` generated during step 1 to authenticate the respective remote peer in a way that establishes or maintains trust. They may be handled similarly to other certificates as they are used to authenticate the respective remote peer. It is conceivable that no subsequent certificate-based authentication is performed considering that the Kyber public keys already establish the authentication.

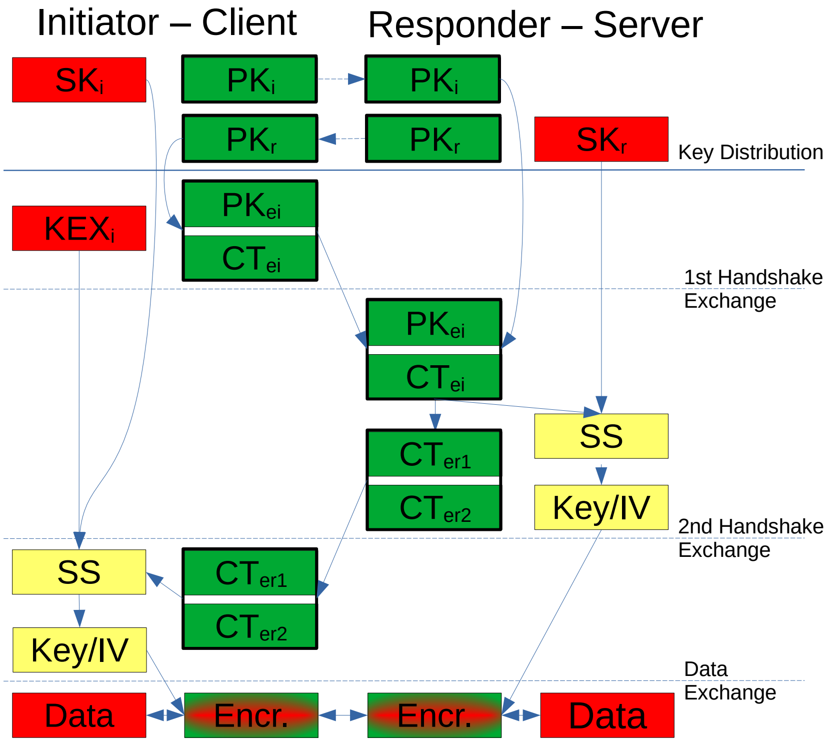

Figure 1: 2-way Kyber KEX

2.1.3 1st Operation: Client - Initiator Handshake

The client has the following information available or generated before:

1. a Kyber keypair `pk_i` and `sk_i` where the public key `pk_i` is registered with the server, and
2. the server's Kyber public key `pk_r`.

The client generates the following information for every handshake anew:

1. perform a Kyber KEX encapsulation initiation operation with `pk_r` and obtain the Kyber KEX data `pk_e_i` and `ct_e_i`

The client sends the following data to the server as part of the initiator handshake:

1. Kyber KEX data `pk_e_i`,
2. Kyber KEX data `ct_e_i`

The red-marked parts of the table are covered by this operation:

Table 2: 1st Operation of Kyber Key Exchange

Step	Alice (Inititiator)	Bob (Responder)
1	Key gen: `pk_i`, `sk_i`	Key gen: `pk_r`, `sk_r`
2	Send public key `pk_i` ;	Send public key `pk_r` ;
	Receipt of `pk_r`	Receipt of `pk_i`
3	Initiate key exchange - generate: ephemeral public key `pk_e_i`, ephemeral ciphertext `ct_e_i`, KEM shared secret `tk`, ephemeral secret key `sk_e`.	
4	Send KEX data `pk_e_i`, `ct_e_i`	Receipt of `pk_e_i`, `ct_e_i`
5		Calculate: ephemeral ciphertext `ct_e_r_1`, ephemeral ciphertext `ct_e_r_2`, shared secret `ss`
6	Receipt of `ct_e_r_1`, `ct_e_r_2`	Send `ct_e_r_1`, `ct_e_r_2`
7	Calculate: Shared Secret `ss`	

2.1.4 2nd Operation: Server - Responder Handshake

The server has the following information available or generated before:

1. a Kyber keypair `sk_r` and `pk_r` where the public key `pk_r` is registered with the client, and
2. the client's Kyber public key `pk_i`.

The server receives the client data and performs the following operations:

1. perform a Kyber KEX responder operation with:

   a. `pk_i`,

   b. the initiator Kyber KEX data, and

   obtain the new Kyber KEX data `ct_e_r_1`, `ct_e_r_2` as well as the shared secret of the required size.
2. Forward the generated shared secret `ss` to the PRF for further processing.
3. destroy Kyber ephemeral shared secret data.

The server sends the following data to the client:

1. Kyber KEX data `ct_e_r_1`,
2. Kyber KEX data `ct_e_r_2`, and
3. a return code of the server's operation.

The red-marked parts of the table are covered by this operation:

Table 3: 2nd Operation of Kyber Key Exchange

Step	Alice (Inititiator)	Bob (Responder)
1	Key gen: `pk_i`, `sk_i`	Key gen: `pk_r`, `sk_r`
2	Send public key `pk_i` ;	Send public key `pk_r` ;
	Receipt of `pk_r`	Receipt of `pk_i`
3	Initiate key exchange - generate: ephemeral public key `pk_e_i`, ephemeral ciphertext `ct_e_i`, KEM shared secret `tk`, ephemeral secret key `sk_e`.	
4	Send KEX data `pk_e_i`, `ct_e_i`	Receipt of `pk_e_i`, `ct_e_i`
5		Calculate: ephemeral ciphertext `ct_e_r_1`, ephemeral ciphertext `ct_e_r_2`, shared secret `ss`

Step	Alice (Inititiator)	Bob (Responder)
6	Receipt of `ct_e_r_1`, `ct_e_r_2`	Send `ct_e_r_1`, `ct_e_r_2`
7	Calculate: Shared Secret `ss`	

2.1.5 3rd Operation: Client - Initiator Handshake Completion

The client receives the server data and performs the following operations:

1. perform a Kyber KEX initiator completion operation with:

    a. the ephemeral initiator Kyber KEX data obtained during the 1st operation,

    and obtain the shared secret of the required size.

2. Forward the generated shared secret `ss` to the PRF for further processing.

3. destroy ephemeral initiator Kyber KEX data obtained during the 1st operation that was used for this handshake.

The red-marked parts of the table are covered by this operation:

Table 4: 3rd Operation of Kyber Key Exchange

Step	Alice (Inititiator)	Bob (Responder)
1	Key gen: `pk_i`, `sk_i`	Key gen: `pk_r`, `sk_r`
2	Send public key `pk_i` ;	Send public key `pk_r` ;
	Receipt of `pk_r`	Receipt of `pk_i`
3	Initiate key exchange - generate: ephemeral public key `pk_e_i`, ephemeral ciphertext `ct_e_i`, KEM shared secret `tk`, ephemeral secret key `sk_e`.	
4	Send KEX data `pk_e_i`, `ct_e_i`	Receipt of `pk_e_i`, `ct_e_i`
5		Calculate: ephemeral ciphertext `ct_e_r_1`, ephemeral ciphertext `ct_e_r_2`, shared secret `ss`
6	Receipt of `ct_e_r_1`, `ct_e_r_2`	Send `ct_e_r_1`, `ct_e_r_2`

Step	Alice (Inititiator)	Bob (Responder)
7	Calculate: Shared Secret `ss`	

2.2 Unilateral Authentication

The protocol outlined section Kyber KEX Key Agreement applies to a bi-directional authentication as both sides, client and server, must exchange their public keys `pk_i` and `pk_r` to support the authentication.

TLS offers also the support for unilateral authentication, i.e. where the server only authenticates to the client but not vice versa. This can be achieved also by using Kyber KEX. In the reference implementation [1], the CRYSTALS Kyber authors not only propose the bi-directional Kyber KEX algorithm, but also the unilateral Kyber KEX algorithm. This is also documented in [3]. This unilateral Kyber KEX algorithm offers the same interfaces and executes the same steps as the Kyber KEX with bi-directional authentication. Thus, it will equally fit into TLS.

With the unilateral Kyber KEX authentication, the responder (i.e. the server) only provides its public key `pk_r` to the initiator (the client). The remaining Kyber KEX protocol is the same where the responder only returns one parameter `ct_e_r` instead of `ct_e_r_1`, `ct_e_r_2` as required by the bi-directional authentication.

This implies that with the unilateral authentication using Kyber KEX, the common TLS approach can still be implemented and yet, the core concern of this analysis, the security strength of the shared secret, is still upheld.

Also the second aspect of this analysis is covered: the responder's public key `pk_r` which is used for the unilateral authentication is cryptographically bound to the Kyber KEX shared secret `ss`.

3 Conclusion

The proposed use of CRYSTALS Kyber KEM as part of the TLS protocol exhibits the issue that the TLS server only contributes to the security strength of the shared secret generated by Kyber. By using the Kyber KEX algorithm which builds on the Kyber KEM approach can be used to alleviate the issue. Using the Kyber KEX algorithm the client and the server both independent of each other contribute to the security of the communication channel where the channel even retains its security when one side has insufficient entropy.

By using the cryptographically connected Kyber-KEX-intrinsic authentication, a subsequent and yet cryptographically disconnected authentication based on signatures can be dropped.

The outlined solution fits directly into the data exchanges that are stipulated by the proposed TLS exchange. It does not require any changes to the protocol at all except for the invocation of Kyber as outlined above.

As the key agreement operations in the SSHv2 and IKEv2 protocols are very similar to TLS, the proposal can be adopted for those protocols as well.

4 References

Draft IETF proposal for integrating Kyber into TLS

[1] CRYSTALS Kyber Reference Implementation

[2] Authenticated Kyber KEX Algorithm Documentation

[3] Unilaterally Authenticated Kyber KEX Algorithm Documentation